Java
Netbeans

اعداد الطالبان:
سليم شرف الشميري
هيثم فكري العواضي

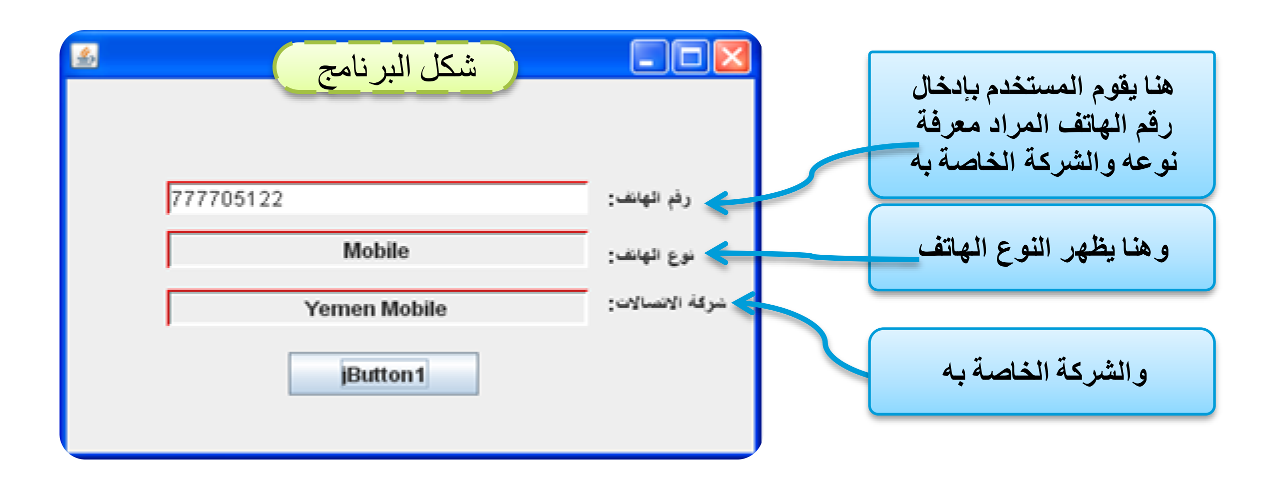
تحت اشراف :-
أ/ ماهر عبد الرحمن

الدوال الجاهزة

اسم الدالة	وظيفتها
Length	لإيجاد طول النص.
CharAt	لاقتصاص رمز من سلسلة نصية.
StartsWith	تستخدم لفحص بداية نص معين وتعيد **True** إذا كان صحيح و **False** إذا كان خطاء.
EndsWith	تستخدم لفحص نهاية سلسلة نصية وتعيد **True** إذا كان صحيح و **False** إذا كان خطاء.
Contiants	تستخدم للبحث عن جزء من النص في سلسلة نصية.
GetChar	تستخدم لقص حرف واحد في سلسلة نصية الى داخل مصفوفة.
Trim	تستخدم لتنظيف أو مسح الفراغات الموجودة على جوانب النص.
IndexOf	لإظهار رقم موقع الحرف المدخل فيها.
CopyValueOf	تنسخ محتويات مصفوفة الى متغير من نوع **String**.

مثال :

أكتب برنامج يقوم المستخدم بإدخال رقم الهاتف ويقوم بإظهار نوع الهاتف والشركة الخاصة به ؟



والكود التالي يكتب داخل **JButton1** في حدث الماوس كليك

قمنا بتعريف متغير من نوع String والقيمة المسندة له هي البيانات التي داخل الأداة JtextField1.

نقوم بالتأكد وفحص قيمة المتغير فإذا كان طوله أكثر من 3 فينفذ ما الأسطر التالية وإلا يعود إلى السطر (17) لتنفيذ الEsle وإظهار رسالة إلى المستخدم بالتأكد من القيمة المدخلة.

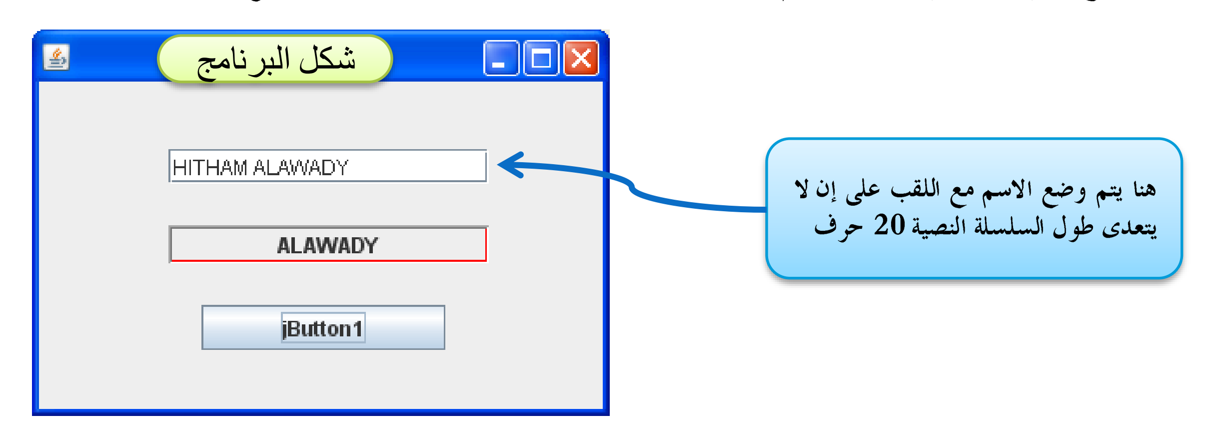
عند تحقق الشرط نقوم بفحص المتغير NO عن طريق الدالة CharAt(0) والتي تقوم بفحص أول حرف للمتغير النصي فإذا تساوى مع العدد 7 فنظهر رسالة بأنه رقم موبايل. والإ فأنه تلفون ثابت.

يقوم هذا الشرط بفحص قيمة المتغير النصي NOفإذا بدء ب77 فتظهر رسالة في lblComp بأن نوع الشركة هي يمن موبايل وهكذا نفس الكود لبقية الشركات.

```
   private void jButton1MouseClicked(java.awt.event.MouseEvent evt) {
1      String No = jTextField1.getText();
2      if (No.length()>= 3) {
3          //charAt
4          if (No.charAt(0)=='7')
5              lblType.setText("Mobile");
6          else
7              lblType.setText("Home");
8          //startsWith
9          if (No.startsWith("73"))
10             lblComp.setText("MTN");
11         else if (No.startsWith("71"))
12             lblComp.setText("SabaPhone");
13         else if (No.startsWith("77"))
14             lblComp.setText("Yemen Mobile");
15         else if (No.startsWith("70"))
16             lblComp.setText("Why");
17     } else {
18         JOptionPane.showMessageDialog(null,"Wrong number");
19     }
```

الاقتصاص

أكتب برنامج يقوم المستخدم بإدخال الاسم واللقب فقط ويقتص اللقب ويضعه في كائن من نوع Jlabel؟



والكود التالي يكتب داخل JButton1 في حدث الماوس كليك

نعرف مصفوفة نصية مكونة من 20 عنصر

```
private void jButton1MouseClicked(java.awt.event.MouseEvent evt) {
1       String Name = jTextField1.getText();
2       if (Name.contains(" ")) {
3           int Start = Name.indexOf(" ") + 1;
4           char ary[] = new char[20];
5               Name.getChars(Start,Name.length(),ary,0);
6               String LName = "";
7               LName = LName.copyValueOf(ary).trim();
8               jLabel1.setText(LName);
9       } else
10          JOptionPane.showMessageDialog(null,"No Last Name");
11  }
```

نفحص المتغير Name فإذا يحتوي على فراغ نفذ السطر (3) والإ يذهب إلى السطر (9) لتنفيذ الـــ Else

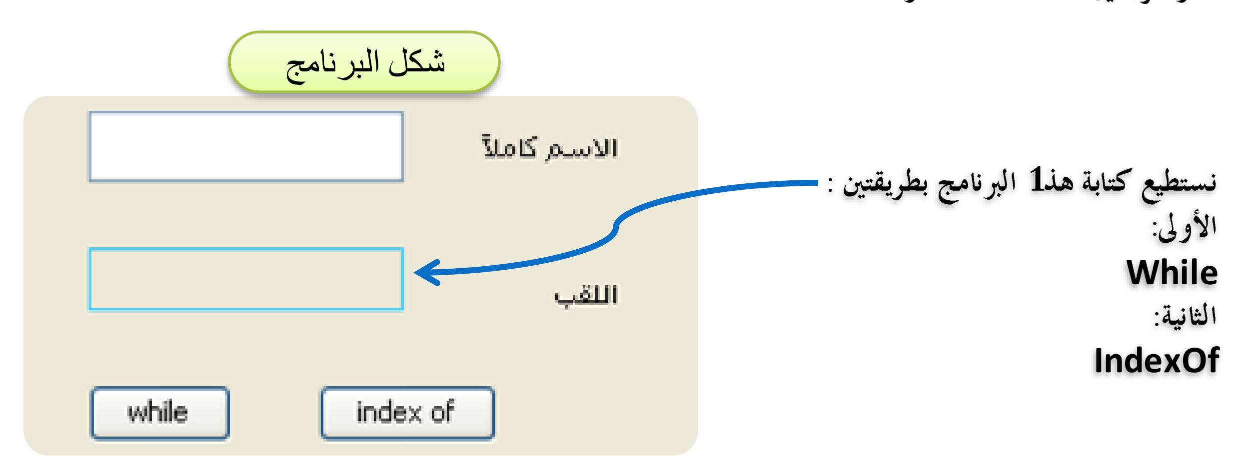
نعرف متغير رقمي ونسند له قيمة المتغير Name ويبدء من الفراغ أي بعد الاسم لأن الاسم مكون من الاسم واللقب فقط حيث سيبدء من الفراغ بعد الاسم + 1.

نقوم باستخدام الدالة GetChar لنقص حرف للمتغير Start من المتغير Name ونضعه في المصفوفة في الموقع صفر (Index) أي أول عنصر. كما درسنا سابقاً بأن أول عنصر في المصفوفة يبدأ من ترقيمه من الصفر.

نسند قيمة للمتغير Lname وبأستخدام الدالة CopyValueOf والتي تقوم بنسخ محتويات المصفوفة إلى المتغير من Lname مع مسح الفراغات الموجودة بداية ونهاية المتغير، وإظهار المتغير في الأداة Jlabel1.

مثال آخر:

أكتب برنامج يقوم فيه المستخدم بإدخال الاسم الرباعي ويقوم باقتصاص اللقب من الاسم ويضعه في كائن من نوع Jlabel وبطريقتين index of أو while؟



نستطيع كتابة هذا1 البرنامج بطريقتين :

الأولى:

While

الثانية:

IndexOf

الكود الخاص بزر index of

متغيير من نوع String وقيمته من الأداة Jtext ونستخدم الدالة Trim لمسح الفراغات التي قد يكتبها المستخدم قبل وبعد كتابة الاسم الرباعي.

```
private void jButton2MouseClicked(java.awt.event.MouseEvent evt) {
1      String n=jTextField1.getText().trim();
2      if(n.contains(" ")) {
3          int s=n.lastIndexOf(" ");
4              String Lname=n.substring(s);
5              jLabel3.setText(Lname);
6          }
7      else
8      {
9          JOptionPane.showMessageDialog(null,"Enter The Last Name");
10     }
}
```

نضع شرط لفحص المتغير، فإذا أحتوى على فراغ نفذ الأسطر التالية، والإ ينهب إلى السطر(7) وتنفيذ ال Else وإظهار رسالة بأنه يجب كتابة اللقب.

نعرف متغير Sمن نوع رقمي، ونسند له قيمة المتغير النصي n والدالة LastIndexOf تبحث عن آخر فراغ في السلسلة النصية وتعيد رقم الموقع للفراغ " " وتعيده قيمة للمتغير s.

نقوم بتعريف متغير LName من نوع String نص ونسند له قيمة والتي هي عبارة عن اقتصاص نص من السلسلة النصية n ونقص منها قيمة المتغير S والذي قد ذكرنا سابقاً بأنه آخر فراغ في السلسلة النصية.
نقوم بإظهار قيمة المتغير LName في الأداة JLabel3 أي اقتصاص اللقب.

الكود الخاص بزر while

```
private void jButton1MouseClicked(java.awt.event.MouseEvent evt) {
    String n=jTextField1.getText().trim();
    if(n.contains(" ")) {
        int s=n.length()-1;
        while(n.charAt(s)!=' ') {
            String Lname=n.substring(s);
            jLabel3.setText(Lname);
            s--;
        }
    }
}
```

والمتغير Sقيمته N.Lenght -1 ولأن الدالة CharAt تبدأ بالترقيم من الصفر والدالة Length تبدأ من الواحد. ولنفترض أن النص عبارة عن 20 حرف فالدالة Length ترقم من 1 إلى 20 ، بينما الدالة CharAt تبدأ بالترقيم من 0 إلى 19 أي عشرين حرف، لذا نساوي بينهم وننقص من المتغير N واحد ليصبح طول المتغير 19 ويتساوى مع CharAt 19.

جملة دوران وشرطها أن يظل يقتص من المتغير S من نهاية الاسم الرباعي حرف في كل دورة من نهاية الاسم كاملاً إلى أن تصبح قيمة المتغير Sمساوية للفراغ وبذلك يقتص الفراغ.

نسند قيمة للمتغير Lname والتي هي اقتصاص من المتغير N حرف واحد في كل دورة من الدوران حتى يتحقق الشرط وتتوقف الجملة عن الدوران عند وجود فراغ في النهاية أي اقتصاص للقب في نهاية السلسلة النصية. ثم نقوم بتنقيص قيمة المتغير S لتصبح طول المتغير N –1 وهكذا يقتص حتى يجد فراغ ويتوقف عن الدوران.
نظهر قيمة المتغير Lname في الأداة Jlabel3.



التشفير

هناك طرق عدة طرق للتشفير وسنقوم هنا بالتشفير بطريقة الزقاق وطريقة عملها كالأتي:

فمثلاً لو لدينا نص مكون من ستة أحرف (Saleem) فأنها أولاً تقوم بتجزئة النص إلى نصفين فيصبح الجزء الأول مكون من (Sal) والجزء الثاني مكون من باقي النص (eem).

ثم نقوم بأخذ أول حرف من الجزء الأول ونأخذ أول حرف من الجزء الثاني لتكون (Se).

وأخذ الحرف الثاني من الجزء الأول والحرف الثاني من الجزء الثاني لتكون (ae).

ونأخذ الحرف الثالث من الجزء الأول والحرف الثالث من الجزء الثاني لتكون (lm).

ونجمع الأجزاء مع بعض ليصبح النص مشفر كالتالي (Seaelm).

ولفك التشفير للنص (Seaelm) نقوم بأخذ أول حرف في النص ونضعه في متغير P1 ونأخذ ثاني حرف ونضعه في متغير آخر P2، ونأخذ ثالث حرف ونضعه في المتغير الأول P1 ونأخذ رابع حرف ونضعه في المتغير P2 وهكذا لبقية الأحرف.

وللتوضيح : النص (Seaelm) كما شرحنا أنه نأخذ أول وثالث وخامس حرف من النص ونضعهم في المتغير P1 ، نأخذ الحروف الثاني والرابع والسادس في المتغير P2.

فيصبح المتغير P1 = (Sal) والمتغير P2 = (eem) ثم نقوم بدمجهم ليصبح النص (Saleem) وبذلك قمنا بفك التشفير للنص.

ملاحظة:

يجب أن يكون النص يقبل القسمة على 2 وبدون باقي أي يكون عدد زوجي وفي حالة كان النص لا يقبل القسمة على 2 أي عدد فردي فأننا نقوم بإضافة أي رمز ليصبح عدد زوجي ونستطيع القيام بعملية التشفير للنص.

مثال:

قم بكتابة برنامج يقوم بتشفير نص في الأداة Jtext1 وإظهار النص المشفر في الأداة Jtext2 ، وكذلك بفك التشفير وعرض النص المشفر في الأداة Jtext3 ؟

شكل البرنامج



الكود التشفير (تشفير)

تعريف متغير T ليأخذ النص المدخل في الأداة Jtext1 من قبل المستخدم ويضعها في المتغير.

```java
private void jButton1MouseClicked(java.awt.event.MouseEvent evt) {
1   String T=jTextField1.getText();
2   if(T.length()%2 !=0)
3       T+=".";
4   int m=T.length()/2;
5   String p1=T.substring(0,m);
6   String p2=T.substring(m);
7   String code="";
8   for(int i=0;i<m;i++)     {
9   code=code+p1.charAt(i)+p2.charAt(i);
10  }
11  jTextField2.setText(code);
    //JOptionPane.showMessageDialog(null
}
```

الأسطر (2و3) نقوم بفحص المتغير فإذا كان يتجزء إلى جزئين كما شرحنا سابقاً فنقوم بتنفيذ الكود في السطر () وإذا لا يتجزء النص إلى جزئين فأننا نقوم بإضافة أي عنصر أو رمز كـــالرمز (.) إلى المتغير حتى تصبح عدد حروف المتغير مساوية لعدد زوجي لتقبل القسمة إلى العدد 2 بدون باقي وعندها نستطيع التشفير.

السطر (4) نعرف متغير **M** لتجزء النص إلى جزئين للتشفير كما شرحنا.

السطر (5) تعريف متغير **P1** ونسند له قيمة ونقتص من المتغير **T** من بدايته إلى النصف.

السطر (6) نعرف متغير **P2** ونسند له قيمة ونقتص من المتغير **T** من النصف إلى النهاية.

السطر (7) تعريف متغير **Code** لتشفير النص داخله.

السطر (8) جملة دوران وتبدأ من الصفر إلى نهاية المتغير **M**.

السطر (9) ونقوم بأخذ أول حرف من المتغير **P1** وأول حرف من المتغير **P2** ونضعهم في المتغير **Code** وتستمر عملية الدوران ونأخذ ثاني حرف من الجزء الأول وثاني حرف من الجزء الثاني ونضعهم في المتغير **Code** وهكذا...... كما شرحنا سابقاً.

السطر (11) إظهار النص المشفر في الأداة **Jtext2** .

الأسطر (1و2و3) نعرف المتغير **T** ويأخذ قيمته من الأداة **Jtext2** النص المشفر, ونعرف المتغيرين **P1,P2**.

الكود فك التشفير (فك التشفير)

```java
private void jButton2MouseClicked(java.awt.event.MouseEvent evt) {
1   String T = jTextField2.getText();
2   String p1 = "";
3   String p2 = "";
4   for ( int i = 0 ; i < T.length(); i+=2 )
5       p1 = p1 + T.charAt(i);
6   for ( int i = 1 ; i < T.length() ; i+=2)
7       p2 = p2 + T.charAt(i);
8   jTextField3.setText(p1+p2);
9   String A = jTextField3.getText();
10  if ( A.endsWith("."))
11      jTextField3.setText(A.substring(0,A.length()- 1));
}
```

الأسطر (4و5) جملة الدوران وتبدأ من الصفر أي من (أول حرف) وشرطها أكبر من طول النص المشفر على أن تكون مقدار الزيادة مرتين (لماذا) ليأخذ أول حرف من النص المشفر وثالث حرف وخامس حرف لأن الـ**i** تزداد بمقدارين حيث يتم تجاهل الحرف الثاني وينتقل إلى الحرف الثالث وهكذا. وتصبح قيمة المتغير **P1** الحروف الأول والثالث والخامس كما شرحنا سابقاً.

الأسطر (6و7) جملة الدوران تبدأ من الواحد أي من (ثاني حرف) وشرطها أكبر من طول النص المشفر وعلى أن تكون مقدار الزيادة مرتين حتى يأخذ الحرف الثاني من النص المشفر ويتجاهل الحرف الثالث ويأخذ الحرف الرابع ويتجاهل الحرف الخامس ويأخذ الحرف السادس. وتصبح قيمة المتغير **P2** الحروف الثاني والرابع والسادس. كما شرحنا سابقاً.

السطر (8) ثم نظهر في الأداة **Jtext3** المتغيرين **P1,P2** وندمجهم.

السطر (9) نعرف متغير من نوع **String** ليأخذ النص الذي تم فك شفرته والتأكد من أنه لا يوجد رمز قمنا بإضافته ليصبح النص قابل للقسمة على 2 بدون باقي.

السطر (10) إذا كان النص ينتهي بالرمز الذي قمنا بإضافته (.)، نقوم باقتصاص الرمز من نهاية النص الذي قمنا بفك شفرته في الأداة **Jtext3** ثم نقوم بإظهار النص بعد قص الرمز ليصبح النص صحيح وكاملاً بدون أي إضافات كما هو موضح في المثال.

الدالة العشوائية Random

تقوم بتوليد أرقام عشوائية أو قيم وتولد كذلك أما **True** أو **False** .

مثال :

– قم بكتابة برنامج يقوم بتوليد أرقام عشوائية في رسالة **Message** على أن تكون الأرقام من الواحد إلى العشرين؟

– أكتب برنامج لإضافة مائة رقم عشوائي إلى مصفوفة من نوع **Object** ثم عرض محتويات هذه المصفوفة إلى كائن من نوع **JlistBox1**؟



كود الرسالة **Message** اختيار رقم عشوائي واحد

```
1    private void jButton1MouseClicked(java.awt.event.MouseEvent evt) {
2 Random x=new Random();
JOptionPane.showMessageDialog(null,""+x.nextInt(100));
    }
```

السطر (1) قمنا بتعريف متغير من نوع عشوائي.
السطر (2) **NextInt** تولد أرقام والقوس الذي بعده نحدد فيه نطاق الأرقام العشوائية التي سظهر للمستخدم وعلى حسب الرقم الموضوع داخل القوس ، وإذا لم نكتب الأقواس أو قيمة فأن الدالة تقوم بتوليد أرقام عشوائية كبيرة.

ملاحظة:

NextBoolean تقوم بتوليد أما **True** أو **False** .

إعداد الطالبان :
سليم شرف ناجي الشميري – هيثم فكري عبد الرقيب العواضي
تحت إشراف الأستاذ:
ماهر عبد الرحمن

كود الإضافة إلى JList1

وضع ارقام عشوائية في LIST

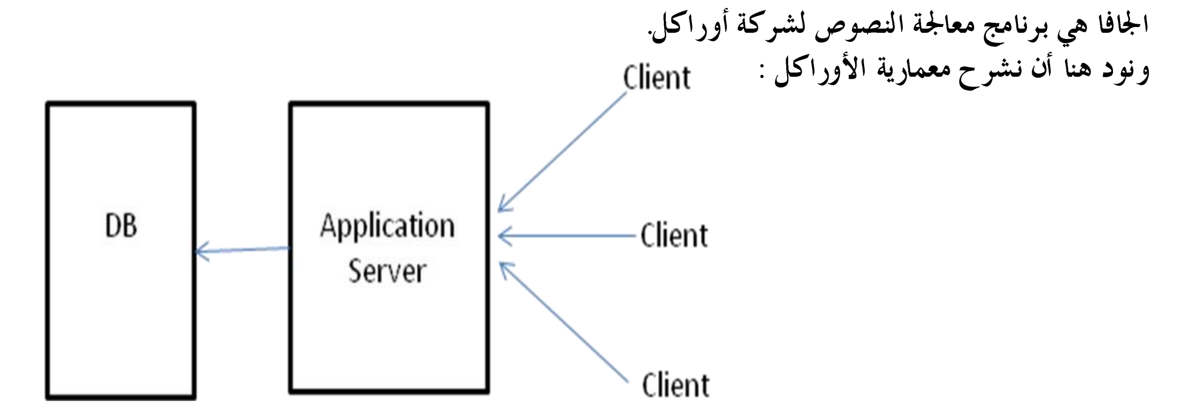
```
private void jButton2MouseClicked(java.awt.event.MouseEvent evt) {
1  Object[] ary =new Object[100];
2  Random x=new Random();
3  for(int i=0;i<100;i++)
4  ary[i]=x.nextInt(100);
5  jList1.setListData(ary);
}
```

السطر (1) نعرف مصفوفة من نوع Object.
السطر (2) نعرف متغير من نوع عشوائي Random.
الأسطر (3و4) جملة الدوران تبدأ من الصفر وتنتهي إلى 100 ونسند قيم للمصفوفة على أن تكون أرقام عشوائية نسبة إلى NextInt.
السطر (5) إظهار قيم المصفوفة الى الكائن Jlist.

JDeveloper

الجافا هي برنامج معالجة النصوص لشركة أوراكل.

ونود هنا أن نشرح معمارية الأوراكل :



تقوم شركة أوراكل بالاشتراك مع شركة Sun تقوم بصنع جهاز Server خاص بالأوراكل فقط ولا نستطيع تحميل أي برامج فيه وهو مصمم فقط لقاعدة بيانات أوراكل وكذلك لتطبيقات الأوراكل.

حيث قاعدة البيانات تكون في جهاز سيرفر والأجهزة الــ Client تأخذ البيانات من تطبيقات السيرفر والتي بدورها تقوم بالتواصل مع قاعدة البيانات للسيرفر وأخذ البيانات المطلوبة.

ولفتح JDeveloper نقوم بالآتي:

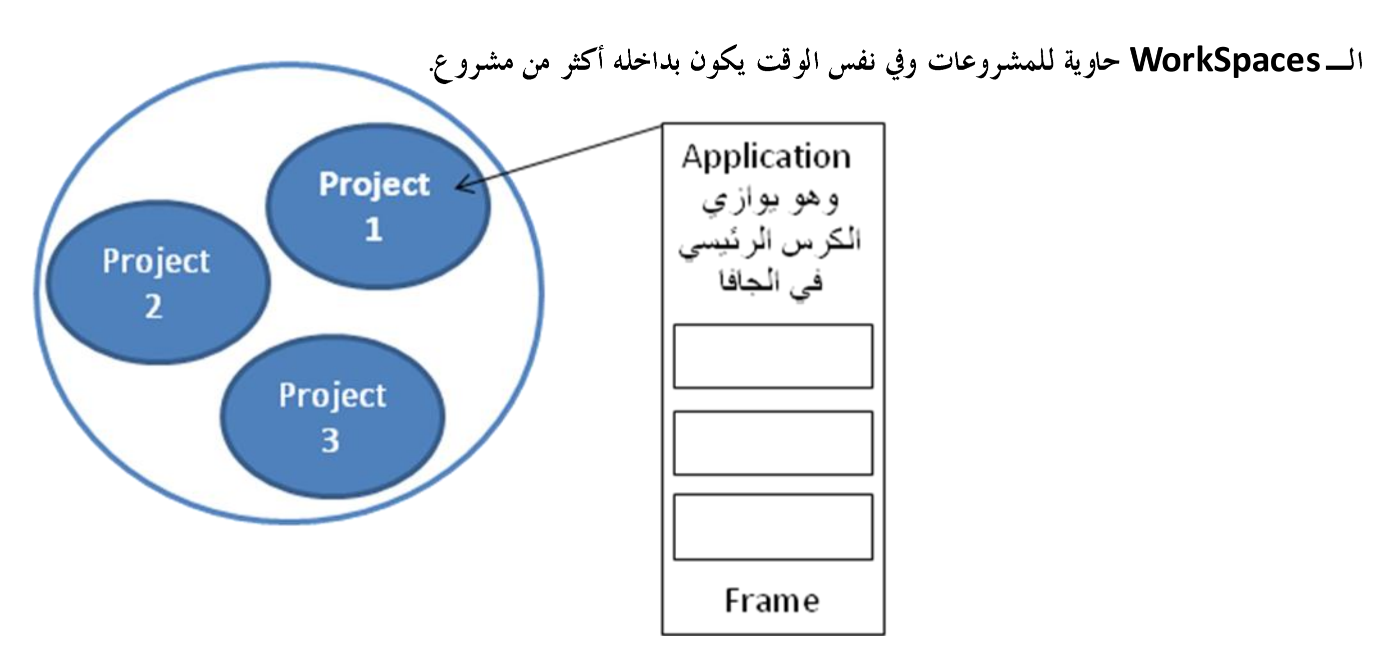
من كافة البرامج نذهب إلى

Oracle Developer Suit – OraDB10g _Home2 ==> JDeveloper ==> J Developer

ولفهم JDeveloper يجب فهم أربعة مصطلحات:

1. WorkSpaces.
2. Project.
3. Application.
4. Frame.

الــ **WorkSpaces** حاوية للمشروعات وفي نفس الوقت يكون بداخله أكثر من مشروع.



لإظهار واجهة التطبيقات نضغط باليمين على **Frame => UI Editor**
وال **UI** تعني **User Interface**.
نطبق على **JDeveloper** نفس الأمثلة التي درسناها في الجافا **NetBeans** في الترم الأول.
ملاحظة:
مكاتب الربط في **JDeveloper** يجب أن نستدعيها كلها، أم في **NetBeans** فأن المعالج يقوم باستدعائها مباشرة ولكن ليس كافة المكاتب أنما يستدعي بعض المكاتب المستخدمة كثيراً.
مثال: مكتبة الربط الخاصة بالرسائل فيجب أن نكتبها في قسم الجنرال الخاص باستدعاء مكاتب الربط

```
Import     Javax.Swing.JoptionPane;
```

هنا سنشرح طريقة عمل الــ **JDeveloper**

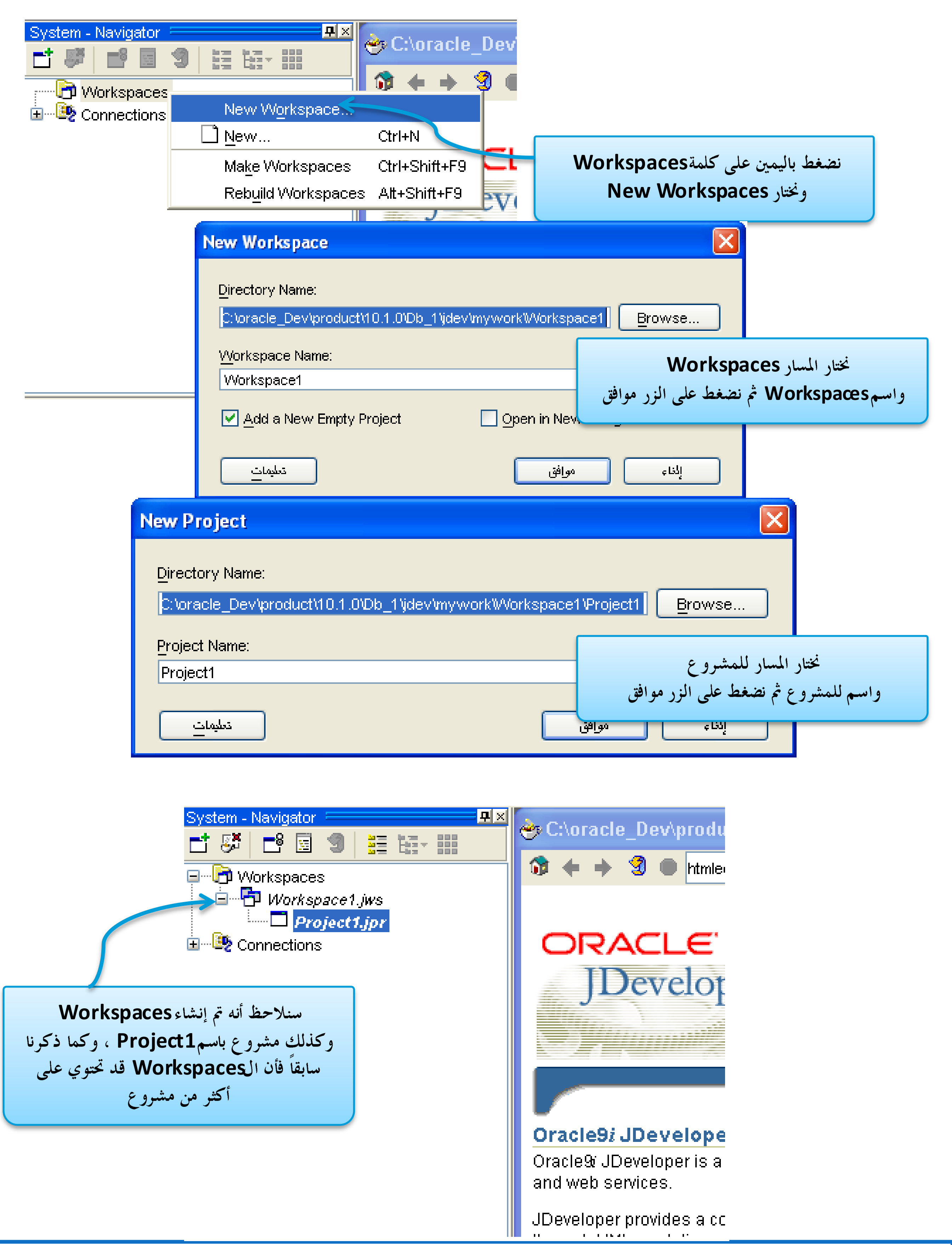
Oracle9i JDeveloper
File Edit Search View Project Run Debug Model Tools Window Help
welcome.html
System - Navigator
Workspaces
Connections
welcome.html - Structure
HTML Structure
C:\oracle_Dev\product\10.1.0\Db_1\jdev\doc\welcome\welcome.html
htmled:/C:/oracle_Dev/product/10.1.0/Db_1/jdev/doc/welcome/welcome.html
ORACLE JDeveloper
JDeveloper | Product Features

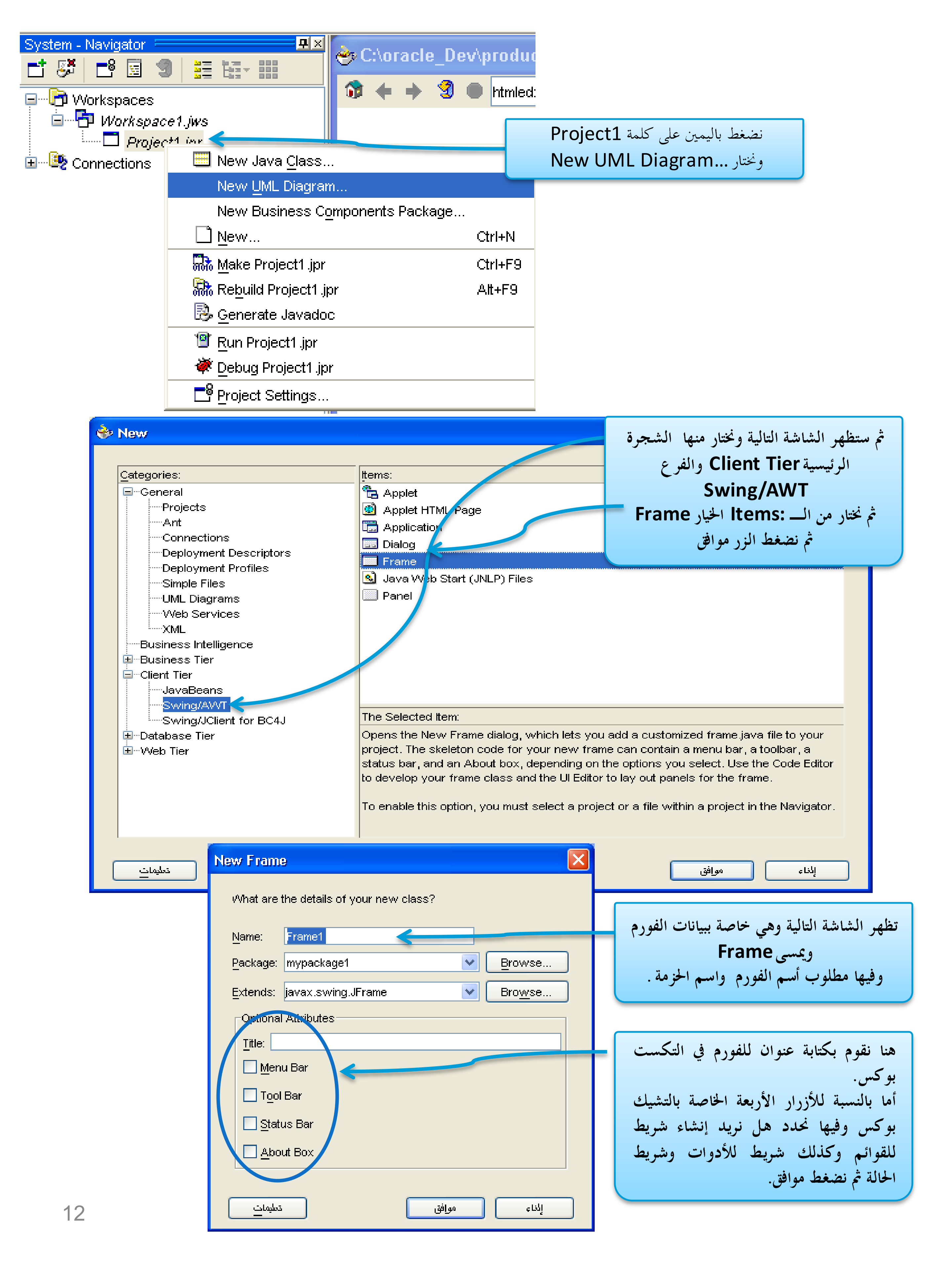
Oracle9i JDeveloper
Oracle9i JDeveloper is a J2EE ™ development environment with end-to-end support for developing, debugging, and deploying e-business applications and web services.

JDeveloper provides a comprehensive set of integrated tools to support the complete development lifecycle, beginning with source control and leading through UML modeling, coding, debugging, testing, profiling, and deploying. JDeveloper simplifies J2EE development by providing wizards, editors, visual design tools, and deployment tools to create high-quality, standard J2EE components, including applets, JavaBeans, JavaServer Pages (JSP), servlets, and Enterprise JavaBeans (EJB). The Oracle Business Components for Java (BC4J) framework further simplifies delivering enterprise applications that adhere to the Model-View-Controller architecture by generating functional business components that implement J2EE design patterns.

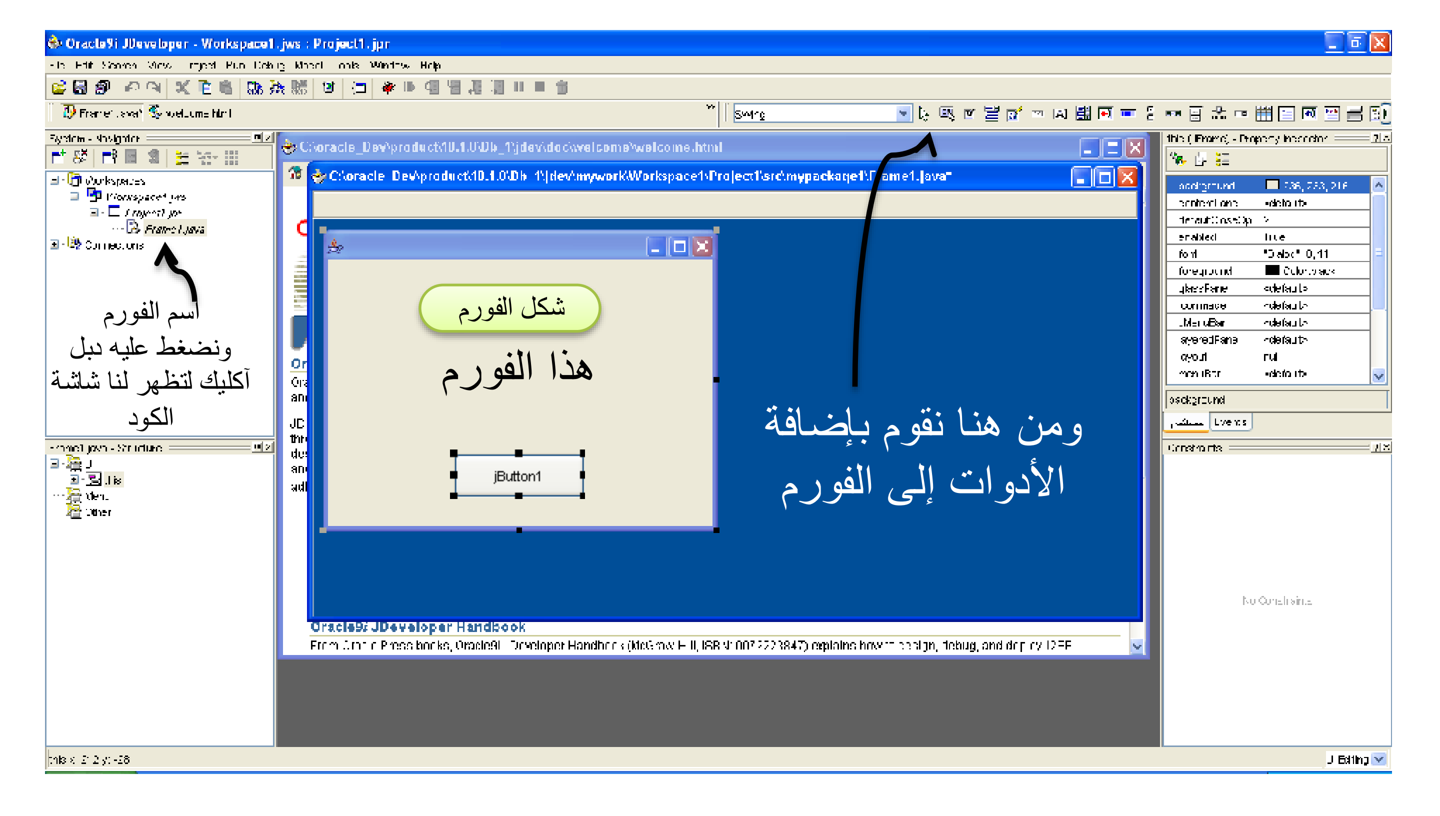
Getting Started
Users to Oracle9i JDeveloper can quickly become familiar with the tool by watching the online demonstrations available on the Oracle Technology Network (OTN). These demonstrations include topics for new users to become familiar with the overall IDE, advanced topics for profiling and deployment, and feature specific topics such as web services and UML modeling. The wide range of topics ensure there is something for every level of developer.

Oracle9i JDeveloper Handbook
From Oracle Press books, Oracle9i JDeveloper Handbook (McGraw-Hill, ISBN: 0072223847) explains how to design, debug, and deploy J2EE
Editing


System - Navigator
Workspaces
Connections
New Workspace...
New... Ctrl+N
Make Workspaces Ctrl+Shift+F9
Rebuild Workspaces Alt+Shift+F9
نضغط باليمين على كلمة Workspaces
ونختار New Workspaces
New Workspace
Directory Name:
C:\oracle_Dev\product\10.1.0\Db_1\jdev\mywork\Workspace1
Browse...
Workspace Name:
Workspace1
Add a New Empty Project
تعليمات
موافق
إلغاء
نختار المسار Workspaces
واسم Workspaces ثم نضغط على الزر موافق
New Project
Directory Name:
C:\oracle_Dev\product\10.1.0\Db_1\jdev\mywork\Workspace1\Project1
Browse...
Project Name:
Project1
نختار المسار للمشروع
واسم للمشروع ثم نضغط على الزر موافق
تعليمات
موافق
إلغاء
System - Navigator
Workspaces
Workspace1.jws
Project1.jpr
Connections
ORACLE
JDeveloper
Oracle9i JDeveloper
Oracle9i JDeveloper is a
and web services.
JDeveloper provides a co
سنلاحظ أنه تم إنشاء Workspaces
وكذلك مشروع باسم Project1 ، وكما ذكرنا
سابقاً فأن الWorkspaces قد تحتوي على
أكثر من مشروع



System - Navigator
Workspaces
Workspace1.jws
Connections
C:\oracle_Dev\produc
htmled:
نضغط باليمين على كلمة Project1
ونختار New UML Diagram...
New Java Class...
New UML Diagram...
New Business Components Package...
New... Ctrl+N
Make Project1.jpr Ctrl+F9
Rebuild Project1.jpr Alt+F9
Generate Javadoc
Run Project1.jpr
Debug Project1.jpr
Project Settings...
New
ثم ستظهر الشاشة التالية ونختار منها الشجرة الرئيسية Client Tier والفرع
Swing/AWT
ثم نختار من الـــ Items: الخيار Frame
ثم نضغط الزر موافق
Categories:
General
Projects
Ant
Connections
Deployment Descriptors
Deployment Profiles
Simple Files
UML Diagrams
Web Services
XML
Business Intelligence
Business Tier
Client Tier
JavaBeans
Swing/AWT
Swing/JClient for BC4J
Database Tier
Web Tier
Items:
Applet
Applet HTML Page
Application
Dialog
Frame
Java Web Start (JNLP) Files
Panel
The Selected Item:
Opens the New Frame dialog, which lets you add a customized frame.java file to your project. The skeleton code for your new frame can contain a menu bar, a toolbar, a status bar, and an About box, depending on the options you select. Use the Code Editor to develop your frame class and the UI Editor to lay out panels for the frame.
To enable this option, you must select a project or a file within a project in the Navigator.
تعليمات
موافق
إلغاء
New Frame
What are the details of your new class?
Name: Frame1
Package: mypackage1
Extends: javax.swing.JFrame
Browse...
Optional Attributes
Title:
Menu Bar
Tool Bar
Status Bar
About Box
تعليمات
موافق
إلغاء
تظهر الشاشة التالية وهي خاصة ببيانات الفورم ويسمى Frame
وفيها مطلوب أسم الفورم واسم الحزمة .
هنا نقوم بكتابة عنوان للفورم في التكست بوكس.
أما بالنسبة للأزرار الأربعة الخاصة بالتشيك بوكس وفيها نحدد هل نريد إنشاء شريط للقوائم وكذلك شريط للأدوات وشريط الحالة ثم نضغط موافق.



C:\oracle_Dev\product\10.1.0\Db_1\jdev\mywork\Workspace1\Project1\src\mypackage1\Fr...

```
package mypackage1;
import javax.swing.JFrame;
import java.awt.Dimension;
import javax.swing.JButton;
import java.awt.Rectangle;

public class Frame1 extends JFrame
{
  private JButton jButton1 = new JButton();

  public Frame1()
  {
    try
    {
      jbInit();
    }
    catch(Exception e)
    {
      e.printStackTrace();
    }

  }
```

Line 1 Column 1 | Insert | Modified | Windows: CR/LF

شكل شاشة الكود وكما نلاحظ وجود بعض المكتبات.
نقوم بكتابة نفس البرامج التي قمنا بكتابتها داخل النت بينز سابقاً والتي درسناها ، ويجب أن لا ننسى أن نستدعي المكتبات الخاصة بالجافا إذ يجب كتابتها في **JDeveloper**

Network

يحتوي كل جهاز الكمبيوتر على 65,535 منفذ (**Port**) والمنافذ من **1023 – 0** خاصة بالويندوز.

:Port

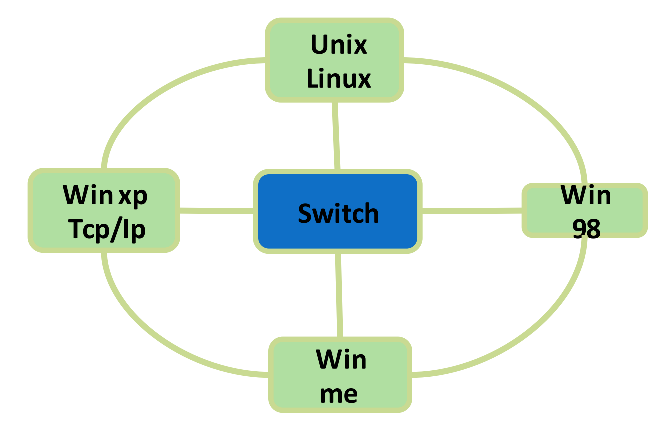
عبارة عن منافذ وهمية للاتصال بالشبكة موجودة داخل كرت الشبكة تستخدم لتشغل البرامج داخل الشبكة في نفس الوقت وبكفاءة دون حصول تعارضات وهو رقم يرسل مع الـــ **FRAM** لتحديد البرنامج المستقبل للرسالة في البرامج، مع العلم أن البورتات ضمن نظام التشغيل عددها من **65535 – 0** ومن المنافذ **1023 – 0** محجوزة للويندوز وما فوق ذلك يمكن استخدامه.

:Socket

هو إطار ترسل فيه البيانات مرفقة بعنوان جهاز الكمبيوتر (**IP**) والبورت (**Port**) لكي لا تضيع البيانات في حالة وجود أكثر من جهاز.

:PROTCOL

هو عبارة عن برنامج **FRAM** مضاف للـــ **IP** للتعرف على الجهاز الهدف ويجمع بين أنظمة التشغيل المختلفة.



:IP

هو عبارة عن عنوان رقمي يميز كل جهاز عن الآخر. مثل (192.168.1.1) ويسمى **IP** وهو عام ويكون في الشبكات الداخلية.

:DNS (DOMIN NAME SERVER)

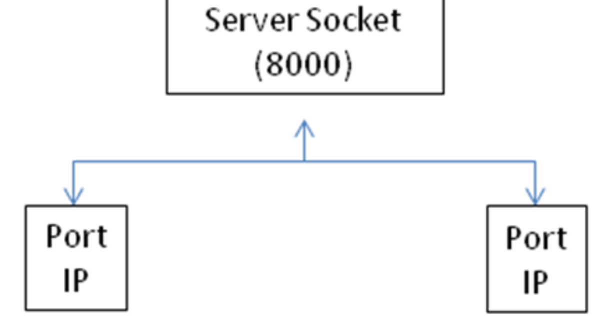
وهو برنامج يقوم بتحويل العنوان المدخل باللغة الانجليزية مثل (WWW.GOOGLE.COM) إلى عنوان **IP** مثل (216.239.61.104) وهو الـــ **IP** الفعلي لجوجل.

:Data Input Stream

وهي عبارة عن البيانات المستقبلة من الأجهزة الأخرى للجهاز الهدف أو الجهاز الرئيسي وتسمى بنهر البيانات القادمة.

:Server Socket

هو الجهاز السيرفر أو المتحكم ببقية الأجهزة ويقوم بفتح بورت فقط ولا يهتم بالأي بي (IP) الخاصة بالاجهزة المتصلة .



طرق القراءة والكتابة من الشبكة:

Data Input Stream.

ويستخدم لاستقبال البيانات من الزبون إلى السيرفر وللإدخال إلى داخل جهاز الكمبيوتر.

Print Stream.

وتستخدم للإخراج (لإخراج التعليمات أو البيانات من السيرفر إلى الزبون).

ملاحظة:

يجب أن نستورد المكتبات الخاصة بالشبكة حينما نتعامل مع الشبكة ونقوم بكتابة مكاتب القراءة والكتابة عبر الشبكة عن طريق استيراد المكتبات وهي كما يلي:

```
Import  Java.net.*;
Import  Java.io.*;
```

ملاحظة:

جميع أوامر التعامل مع الشبكات يجب أن تعمل مع **Try / Catch**.

Thread

Thread : هو الذي يحدد سير المهام المتجهة نحو المعالج.

وهو نوعان :

1. Multi Thread Application.

وتعني البرامج التي تنفذ أكثر من مهمة في نفس الوقت مثل : الألعاب.

2. Single Thread Application.

وهي البرامج التي تنفذ المهام بشكل متسلسل أي تنتهي بالمهمة الأولى ثم المهمة الثانية وهكذا.

أنواع Thread من حيث الاستخدام:

1. Demo Thread.

وهي العمليات التي تتم بدون تدخل من المستخدم أي تتم عن طريق نظام التشغيل أو البرامج.

2. User Thread.

وهي العمليات التي يقوم بها المستخدم.

ملاحظة:

هنالك أولويات لتنفيذ الـــ Thread ولها خاصية وتأخذ من الرقم واحد إلى الرقم عشرة فقط.

وهنا سنقوم بشرح بعض خصائص الـــ Thread :

الخاصية	الشرح
SetPriorpity	وهي لتحديد أولويات المهمة وكلما كان الرقم أكبر كانت الأولوية أعلى أي أن الرقم واحد هو الأقل أولوية والرقم عشرة له أكثر أولوية في التنفيذ وإذا لم تحدد الأولوية فتكون الأولوية الافتراضية هو الرقم خمسة .
SetName	وتستخدم لوضع اسم للـــ Thread.
GetName	وتستخدم لأخذ أو إظهار اسم الـــ Thread.

مثال:

```
Thread1.setName ("Saleem");
JOptionPane.ShowMessageDialog(Null,"New Name : " + Thread1.GetName());
```

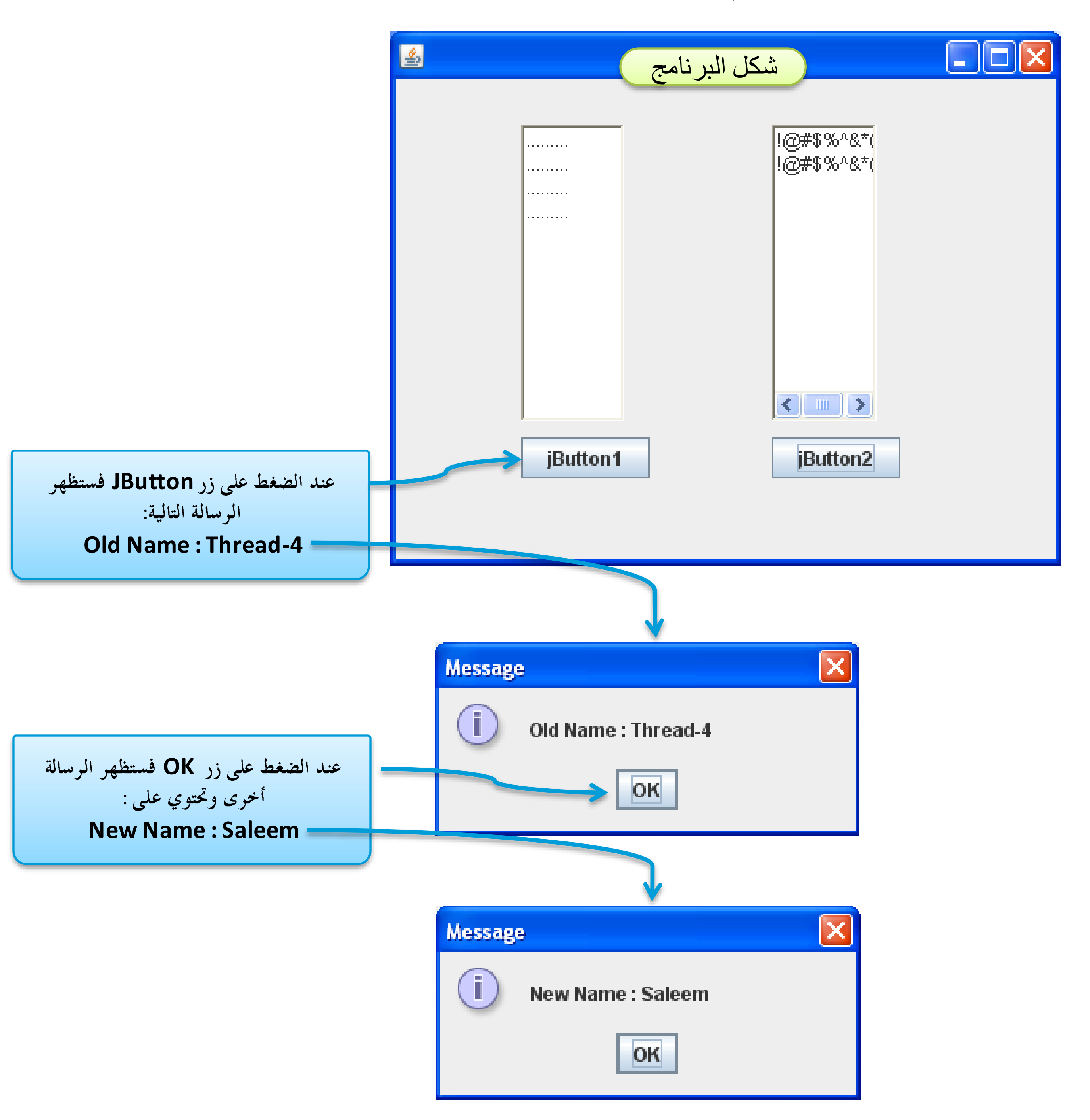
Sleep (1000) :

أي تعني إيقاف التنفيذ المبرمج بمقدار الرقم الممر إلى الدالة Sleep أي بالملي ثانية حيث أن 1000 ملي ثانية تساوي ثانية واحدة.

وتكتب ال Sleep داخل Try/Catch.

مثال :

قم بكتابة برنامج يقوم بتعبئة كائن من نوع **List1** عند الضغط على **Jbutton** يقوم بإظهار رسالة باسم الــ **Thread** القديم ثم بعد الضغط على زر **Ok** في الرسالة الظاهرة في الشاشة يقوم بإظهار رسالة آخرى تحتوي على اسم الــ**Thread** الجديد. وعند الضغط على زر **JButton2** فأنه يقوم بطباعة رموز على الكائن الآخر **List2** ؟



والآن سنقوم بكتابة وشرح الكود :

```
1  public class Thread1 extends Thread {
2      public void run() {
3          while(true) {
4              list1.addItem(".........");
5              try {
6                  this.sleep(1000);
7              } catch(Exception e) {
8              }
9          }
10     }
11 }
```

السطر (1):
قمنا بتعريف كلاس باسم **Thread1** في قسم التصريحات وعلى أن يكون عام وعبارة **extends** تعني أن يورث هذا ال**Thread1** من ال**Thread** الموجود ضمن كلاس داخل لغة الجافا.

السطر (2):
قمنا بتعريف دالة من نوع عام (**Public**) داخل الكلاس الذي قمنا بإنشائه وسميناها **Run()** .

السطر (3):
جملة دوران **while** طالما أن ال**Thread** يعمل أي **True** قم بالتالي.

السطر (4):
إضافة عناصر في كل مرة دوران إلى الكائن **List1** حتى تصبح ال **Thread** تساوي **False** وإلا ستظل العناصر تضاف إلى ما لا نهاية.

السطر (6):
وتعني أن العناصر تضاف إلى الكائن **List1** كل 1000 ملي ثانية أي كل ثانية.

```
1  public class Thread2 extends Thread {
2      public void run() {
3          while(true) {
4              list2.addItem("!@#$%^&*()");
5              try {
6                  this.sleep(2000);
7              } catch(Exception e) {
8              }
9          }
10     }
11 }
```

سبق وشرحنا السطور (1 و 2 و 3).

السطر (4):
إضافة رموز في كل مرة دوران إلى الكائن **List2** حتى تصبح ال **Thread** تساوي **False** وإلا ستظل العناصر تضاف إلى ما لا نهاية.

السطر (6):
وتعني أن العناصر تضاف إلى الكائن **List1** كل 2000 ملي ثانية أي كل ثانيتين.

```
1  private void jButton2MouseClicked(java.awt.event.MouseEvent evt) {
2      Thread2 t2=new Thread2();
3      t2.setPriority(8);
4      t2.start();
   }
```

jButton2

السطر (2):
قمنا بتعريف متغير من النوع الـــ **Thread 2** الذي قمنا بإنشاءه.

السطر (3):
قمنا بإعطاء أولوية رقم (8) للـــ **t2** الذي عرفناه من الـــ **Thread2**.

السطر (3):
قمنا بتشغيل ال**t2**.
أي بعد الضغط على زر **JButton2** سيقوم باستدعاء الـــ**Thread2** والذي يقوم بتعبئة الكائن **List2** بالرموز الموجودة في الكود السابق.

jButton1

```
1  private void jButton1MouseClicked(java.awt.event.MouseEvent evt) {
2      Thread1 t1=new Thread1();
3      t1.setPriority(1);
4      JOptionPane.showMessageDialog(null,"Old Name : " + t1.getName());
5      t1.setName("Saleem");
6      JOptionPane.showMessageDialog(null,"New Name : " + t1.getName());
7      t1.start();
   }
```

سبق وشرحنا السطور (2و 3و7).

السطر (4 و6):
رسالة تقوم بإظهار اسم الـــ **Thread** المسمى **t1** والخاصية الخاصة بذلك هي**GetName**.

السطر (5):
الخاصية **SetName** وظيفتها تغير اسم ال **Thread** المسمى **t1** إلى أي اسم نريده.

مثال:

قم بكتابة برنامج عبارة عن ساعة إيقاف عن طريق الـــ **Thread** وتحتوي على زر للتشغيل وآخر للإيقاف وآخر لتصفية الشاشة ؟

شكل البرنامج

0 : 2 : 5

Start Stop Clear

خطوات الحل:

1– نقوم باستدعاء المكتبة التالية:

Import org.omg.SendingContext.RunTime;

2– نقوم بتكوين **Thread** في قسم التصريحات ونسميه **Stpowatch** .

تم شرح السطور (1 و 4 و 9):

السطر (2 و 3):

قمنا بتعريف متغير من نوع منطقي وجعلنا قيمته تساوي **False** حتى لا يظل يعمل الـــ **Thread** لأنه يتوقف عندما تكون قيمة الدالة **Run()** تساوي **false** ،

وقمنا بتعريف ثلاثة متغيرات من نوع رقمي والتي تدل على أن **S** تعني ثانية و **m** تعني دقيقة و **h** تعني ساعة.

```
1  public class stopwatch extends Thread {
2      boolean Run = false;
3      int s,m,h;
4      public void run() {
5          while(Run){
6              jLabel1.setText(""  +  h   +  " : "   +   m  +  " : "  +   s );
7              s++;
8              try {
9                  this.sleep(10);
10                 if (s >= 59  ) {
11                     s=0;
12                     m++;
13                 } else if( m >= 59) {
14                     h++;
15                     m=0;
16                 } else if(h >= 12) {
17                     h=0;
18                 }
19             }catch (Exception e) {
20                 JOptionPane.showMessageDialog(null,e.getMessage());}
21         }
22     }
23 }
```

نقوم هنا بشرح طريقة عمل الساعة وتحويل ذلك إلى كود:

السطر (6):

إظهار المتغير **h + m + s** وبينهم العلامة ":".

السطر (10 و11 و12):

وضعنا شرط للثواني حيث أذا كانت أقل أو تساوي 59 قم بجعل الثواني تعود إلى صفر ونقوم بزيادة الدقائق حتى تزيد بعد مرور 60 ثانية في كل مرة.

السطر (13):

إذا وصلت الدقائق إلى 59 نقوم بجعل المتغير الخاص بالساعة ونزيده في كل مرة تصل فيها الدقائق إلى 59 دقيقة، ونجعل الدقائق ترجع وتعود إلى الصفر.

```java
private void jButton3MouseClicked(java.awt.event.MouseEvent evt) {
    ws.Run=false;
    jLabel1.setText("0");
}
```

Clear

في هذا الزر قمنا بإيقاف عمل المتغير ws المورث من الكلاس الذي تم إنشاءه والمسمى stopwatch وجعل القيم داخل الكائن Jlabel1 تساوي للصفر.

```java
private void jButton2MouseClicked(java.awt.event.MouseEvent evt) {
    ws.Run=false;
}
stopwatch ws;
private void jButton1MouseClicked(java.awt.event.MouseEvent evt) {
    ws=new stopwatch();
    ws.s= Integer.parseInt(jLabel1.getText());
    ws.m= Integer.parseInt(jLabel1.getText());
    ws.h= Integer.parseInt(jLabel1.getText());
    ws.Run=true;
    ws.start();
}
```

Stop

نجعل قيمة المتغير مساوية لـــ false والتي تقوم بإيقاف الوقت.

Start

في السهم العلوي قمنا بتعريف المتغير WS في قسم التصريحات والمتغير من نوع كلاس stopwatch.

في السطر التالي قمنا بجلب قيمة المتغير S الثواني من الكائن Jlabel1 وكذلك للمتغير M , H الخاص بالدقائق والساعات.

ثم تفعيل عمل المتغير ws عبر جعل قيمة التشغيل له مساوية True.

ثم بداية عمل المتغير ws.

Chat

وبعد أن ذكرنا بعض التعريفات المتعلقة بالشبكة سوف نقوم الآن بشرح طريقة عمل الاتصال بين جهازين عبر الشبكة.

مثال:

أكتب برنامج لعمل شات بين جهازين يكون احدهما رئيسي و الأخر فرعي ؟

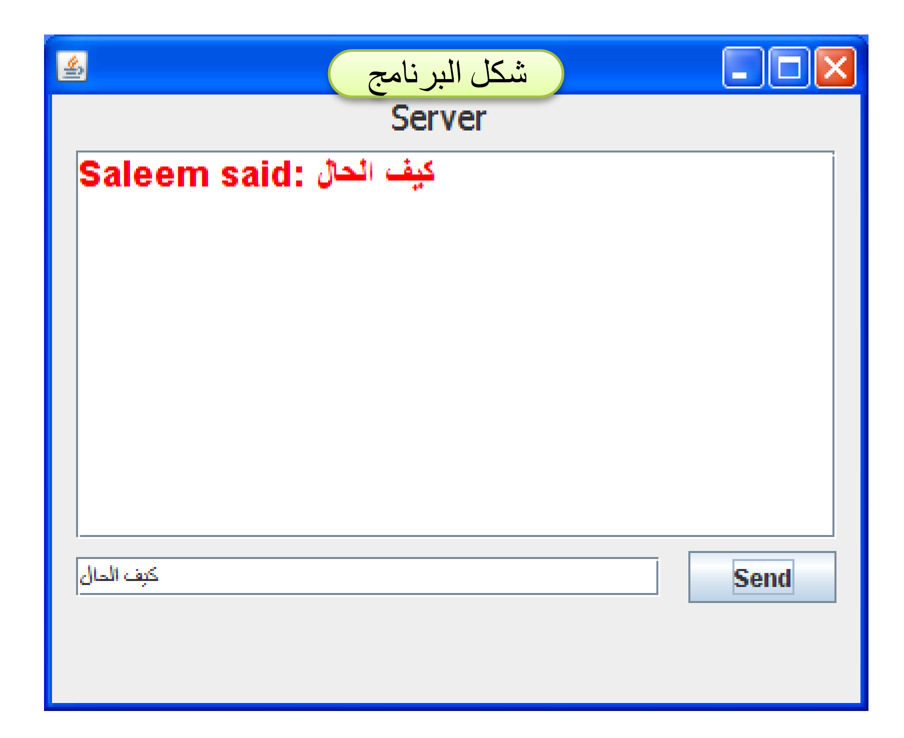
نقوم بتصميم الشاشة التالية ونضع فيها الأدوات التالية:

في واجهة السرفر:

1. Jlabel كائنين.
2. Jbutton كائن واحد.
3. jTextArea1 كائن واحد.
4. JTextField كائن واحد.

في واجهة الأخرى:

1. Jlabel كائنين.
2. Jbutton كائن واحد.
3. jTextArea1 كائن واحد.
4. JTextField كائن واحد.





شاشة السرفر

```
import java.awt.Color;
import java.awt.Font;
import java.net.*;
import java.io.*;
import javax.swing.JOptionPane;
```

يجب علينا أولاً استيراد المكتبتين الخاصة بعملية القراءة والكتابة عبر الشبكة وهما :

Import java.net.*;
Import java.io.*;

```
1 ServerSocket serverS;
2 Socket soc;
3 DataInputStream in;
4 PrintStream out;
```

في قسم التصريحات نعرف الآتي:

السطر (1): قمنا بتعريف متغير من نوع **ServerSocket** ليكون الجهاز هذا سيرفر أو المتحكم ببقية الأجهزة.

السطر (2): قمنا بتعريف متغير **soc** لنحدد أذا كان سيرفر اما لا في الاكواد التالية.

السطر (3 و4): تعريف متغيرين الأول **in** لعملية القراءة عبر الشبكة أي الإدخال والثاني **out** لعملية الكتابة عبر الشبكة أي الإخراج.

```
1  public class connectth extends Thread {
2      public void run() {
3          try {
4              serverS = new ServerSocket(9000);
5              soc = serverS.accept();
6              in = new DataInputStream(soc.getInputStream());
7              out = new PrintStream(soc.getOutputStream());
8              jLabel1.setText("Connected.....");
9          } catch(Exception e){
10             JOptionPane.showMessageDialog(null,e.getMessage());
11         }
12     }
13 }
```

في السطر (1): أنشئنا كلاس من نوع **Thread** وقمنا بتسمية **connectth** ليقوم بعملية الاتصال عبر الشبكة.

السطر (2): دالة داخلة الكلاس من نوع عام واسمها **run** .

السطر (4): اسندنا قيمة للمتغير **serverS** وجعلنا البورت الخاص به يساوي **9000** كما ذكرنا سابقاً بأن نظام التشغيل يحتوي على 65535 بورت فقمنا بإختيار هذا البورت ويمكن أن نضع أي رقم من بعد 1023 وذلك لأن البورتات من صفر إلى 1023 خاصة بالويندوز.

السطر (5): جعل المتغير **soc** على أن يكون سرفر. وحيث أن بقية الأجهزة المرتبطة بالسرفر يجب أن يكون لها **IP** خاصة لكل جهاز للتعرف عليها ضمن الشبكة ، ما عدا جهاز السرفر فليس له **IP** للتعرف عليه.

السطر (6): نسند قيمة للمتغير **in** ليقوم بإحضار البيانات عبر الشبكة.

السطر (7): نسند قيمة للمتغير **out** ليقوم بإخراج البيانات عبر الشبكة.

السطر (8): إظهار كلمة **......Connected** في حالة أن الاتصال عبر الشبكة يعمل والا يذهب إلى الــ **Catch** لينفذ السطر لجلب الخطاء.

```
1  private void jButton1MouseClicked(java.awt.event.MouseEvent evt) {
2      jTextArea1.setFont(new Font("Arial",Font.BOLD,18));
3      jTextArea1.setForeground(Color.red);
4      String m = jTextField1.getText();
5      jTextArea1.append("Saleem said: " + m + "\n" );
6      out.println( m );
7      jTextField1.setText("");
8      jTextField1.requestFocus();
9  }
```

السطر (2): الكائن **JTextArea1** تستخدم للتخاطب عبر الشات وهنا قمنا بوضع الإعدادات الخاصة الخط.

السطر (3): لون الخط داخل الكائن **JTextArea1** .

السطر (4): تعريف متغير لجلب البيانات من الكائن **JTextField**.

السطر (5): الخاصية **append** وتقوم بجلب البيانات الموجودة داخل المتغير **m** وعدم مسح الكائن **JTextArea1** بالبيانات السابقة وإنما إضافة بيانات في سطر جديد.

السطر (6): إخراج البيانات الموجودة داخل المتغير **m** عبر الشبكة ليستقبلها الجهاز الأخر.

السطرين (7 و 8): تفريغ الكائن **JTextField** وجعله فارغ وإعادة تركيز مؤشر الماوس على الكائن **JTextField** .

```
private void formWindowOpened(java.awt.event.WindowEvent evt) {
    connectth cth = new connectth();
    cth.start();
}
```

تعريف متغير **cth** من النوع **connectth** وبداية عمل المتغير داخل الحدث **formWindowOpened** حتى يعمل الاتصال عند فتح الفورم.

شاشة الـ Client

يجب علينا أولاً استيراد المكتبتين الخاصة بعملية القراءة والكتابة عبر الشبكة وهما :

Import java.net.*;
Import java.io.*;

كما شرحنا سابقاً نعرف المتغيرات **soc** و **in** و **out** .

```
Socket soc;
DataInputStream in;
PrintStream out;
```

```
1  private void getConnection() {
2      try {
3          soc = new Socket("127.0.0.1",9000);
4          in = new DataInputStream(soc.getInputStream());
5          out = new PrintStream(soc.getOutputStream());
6          jLabel2.setText("Connecting");
7      }catch(Exception e){
8          JOptionPane.showMessageDialog(null,e.getMessage());
9      }
10 }
```

نكتب دالة من نوع خاص ونسميها **getConnection** وهي لتقوم بجلب الاتصال عبر الشبكة.
السطر (3): اسندنا قيمة للمتغير **soc** وجعلنا البورت الخاص به يساوي **9000** كما ذكرنا سابقاً وكذلك نحدد الــ **IP Adress** الخاص بالجهاز الــ **Client** ونضع له البورت الخاص بجهاز السرفر (**9000**).
الأسطر (4 و 5): قد تم شرحها سابقاً.

إعداد الطالبان :
سليم شرف ناجي الشميري – هيثم فكري عبد الرقيب العواضي
تحت إشراف الأستاذ:
ماهر عبد الرحمن

```java
1  private void reciveMessage() {
2      try {
3          //jTextArea1.setFont(new Font("Arial",Font.ITALIC,14));
4          jTextArea1.setForeground(Color.BLUE);
5          String m = in.readLine();
6          if(m.length()>1)
7              jTextArea1.append("Basheer said: " + m + "\n" );
8      } catch(Exception e) {
9          JOptionPane.showMessageDialog(null,e.getMessage());
10     }
11 }
```

نكتب دالة من نوع خاص ونسميها **reciveMessage** وهي لتقوم بإستقبال الرسائل عبر الشبكة.
السطر (4): الكائن **jTextArea1** نجعل لون الخط أزرق.
السطر (5): تعريف متغير **M** من نوع نصي ونسند له قيمة لقراءة المتغير **in** المعرف في قسم التصريحات والذي يقوم بقراءة البيانات عبر الشبكة ولأن البيانات المرسلة عبر عن نص ، لذلك عرفنا المتغير **m** ليحتفظ بالنصوص المرسلة عبر الشبكة داخل هذا المتغير.
السطر (6 و 7): وضعنا شرط حيث أذا كان النص المرسل أكبر من واحد. وفي حالة تحقق الشرط يتم تنفيذ السطر (7) والذي سبق وشرحناه.

```java
private void formMouseClicked(java.awt.event.MouseEvent evt) {
    reciveMessage();
}
```

نستدعي الإجراء **reciveMessage();** داخل الحدث الخاص بمرور مؤشر الماوس على الفورم.

```java
private void formWindowOpened(java.awt.event.WindowEvent evt) {
    getConnection();
}
```

نستدعي الإجراء **getConnection();** داخل الحدث الخاص بفتح نافذة الفورم.

مثال آخر على القراءة والكتابة عبر الشبكة:
أكتب الكود اللازم لعمل برنامج مكون من مشروعين الأولى عبارة عن جهاز يتحكم ويتجسس على المشروع الثاني والذي هو عبارة عن جهاز الضحية ؟





خطوات الحل :

أولاً : بإنشاء مشروع ونسمي الفورم **FrmServer**.

ثانياً : نقوم بإنشاء مشروع آخر ونسمي الفورم **FrmClient**.

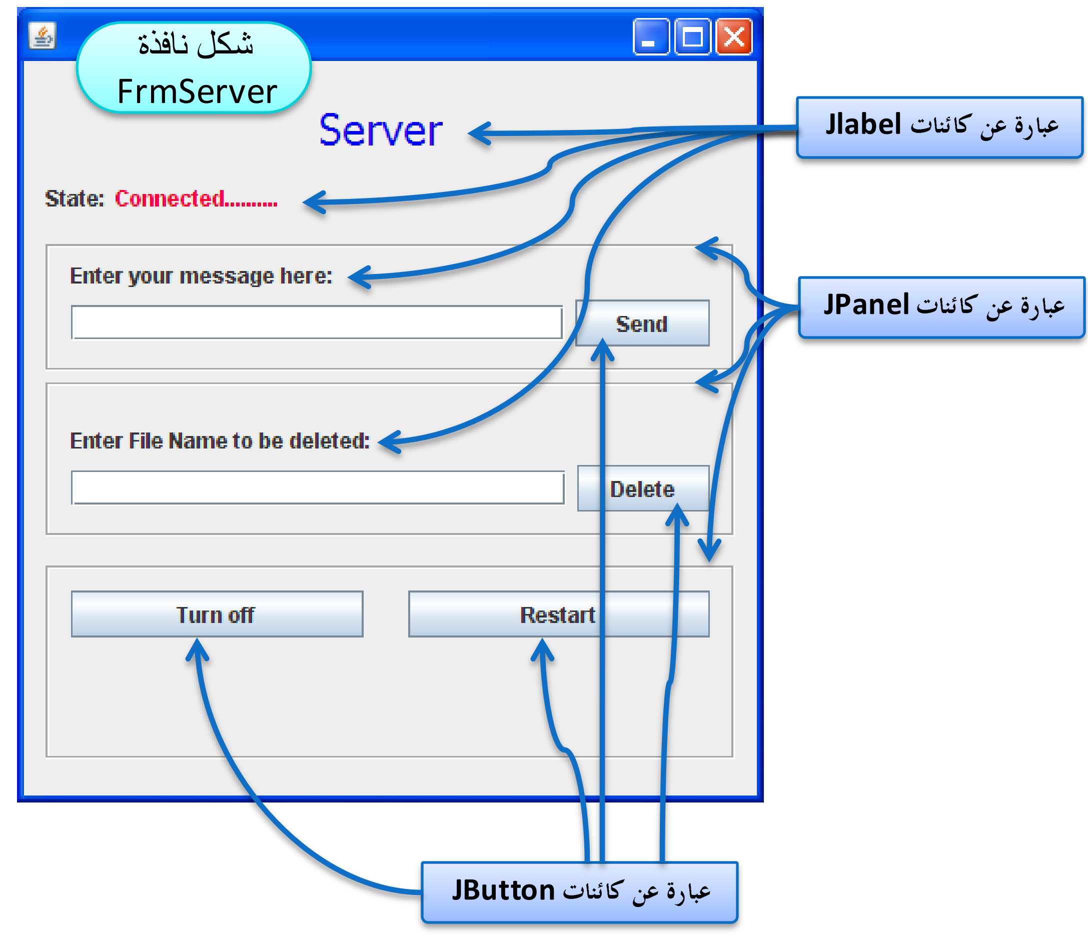
أولاً : في المشروع **FrmServer** نضع الأدوات التالية:

الكائن	الاسم البرمجي	الاسم الظاهري
Jlabel1	lblState	SERVER
Jlabel2		State
Jlabel3		Not Connect
JPanel1 عبارة عن حاوية نضع فيه الأدوات التالية:		
Jlabel4		Enter Your Message Here:
JTextField1		
JButton1	btnSend	Send
JPanel2 حاوية نضع فيه الأدوات التالية:		
Jlabel5		Enter Your Name To Be Deleted:
JTextField2		
JButton2	btnDelete	Delete
JPanel3 حاوية نضع فيها الأدوات التالية:		
JButton3	btnRestart	Restart
JButton4	btnTurnoff	Turn Off

ثانياً : في المشروع **FrmClient**:

لا نقوم بوضع أي شي على الفورم لأننا فقط نريد أن نتجسس على جهاز الضحية ، لذا لا نضع أدوات وحتى أننا نقوم بإخفاء الفورم حتى لا يعلم الضحية بوجود نافذة مفتوحة، وسيأتي شرح ذلك لاحقاً.

أولاً : شاشة FrmServer



<u>سنقوم الآن بشرح الأكواد الخاصة بالمشروع **FrmSerevr**:</u>

```
1  import java.net.*;
2  import java.io.*;
3  import javax.swing.JOptionPane;
4
5  public class frmServer extends javax.swing.JFrame {
6
7      public frmServer() {
8          initComponents();
9      }
10
11     ServerSocket ServerS;
12     Socket soc;
13     DataInputStream in;
14     PrintStream out;
```

السطور (1 و 2):
في قسم التصريحات نقوم باستدعاء المكاتب الخاصة بعملية القراءة والكتابة عبر الشبكة.

ثم في قسم التصريحات نعرف كذلك الآتي:
السطر (11): قمنا بتعريف متغير من نوع **ServerSocket** ليكون الجهاز هذا سيرفر أو المتحكم ببقية الأجهزة الموجودة عبر الشبكة.
السطر (12): قمنا بتعريف متغير **soc** لنحدد أذا كان هذا الجهاز سيرفر ام لا في الأكواد التالية.
السطر (13و14): تعريف متغيرين الأول **in** لعملية القراءة عبر الشبكة أي الإدخال، والثاني **out** لعملية الكتابة عبر الشبكة أي الإخراج.

في السطر (1): أنشئنا كلاس من نوع Thread وقمنا بتسمية connectth ليقوم بعملية الاتصال عبر الشبكة.
السطر (2): دالة داخلة الكلاس من نوع عام واسمها run .

```
1  public class ConnectTh extends Thread {
2      public void run() {
3          try{
4              ServerS = new ServerSocket(9000);
5              soc = ServerS.accept();
6              in = new DataInputStream(soc.getInputStream());
7              out = new PrintStream(soc.getOutputStream());
8              lblState.setText("Connected..........");
9              btnSend.setEnabled(true);
10             btnDelete.setEnabled(true);
11             btnTurnOff.setEnabled(true);
12             btnRestart.setEnabled(true);
13         }catch(Exception e){
14             JOptionPane.showMessageDialog(null,e.getMessage());
           }
       }
   }
```

السطر (4): اسندنا قيمة للمتغير serverS وجعلنا البورت الخاص به يساوي 9000 كما ذكرنا سابقاً بأن نظام التشغيل يحتوي على 65535 بورت فقمنا بإختيار هذا البورت ويمكن أن نضع أي رقم من بعد 1023 وذلك لأن البورتات من صفر إلى 1023 خاصة بالويندوز .

السطر (5): جعل المتغير soc على أن يكون سرفر. وحيث أن بقية الأجهزة المرتبطة بالسرفر يجب أن يكون لها IP خاصة لكل جهاز للتعرف عليها ضمن الشبكة ، ما عدا جهاز السرفر فليس له IP للتعرف عليه.

السطر (6): نسند قيمة للمتغير in ليقوم بإحضار البيانات عبر الشبكة.
السطر (7): نسند قيمة للمتغير out ليقوم بإخراج البيانات عبر الشبكة.

السطر (8): نجعل الاسم الظاهري للكائن JLabel1 بأسم "Connected" أي أنه في حالة تم الاتصال عبر الشبكة فأنه يظهر بأنه متصل وإلا فأنه سيظل كما تم تسميته سابقاً ب "Not Connect" .
السطور (9و10و11و12): نقوم بجعل الكائنات الموجود على الفورم تعمل حيث الكائنات JButton قبل أن يكون هنالك إتصال بالفورم فأن الخاصية لها Enabled مساويه لـــ False أي غير مفعلة .
ولكن بعد عملية الاتصال عبر الشبكة فأن الكائنات JButton تصبح مفعلة.

كود Restart

```
1 private void btnRestartMouseClicked(java.awt.event.MouseEvent evt) {
2     if(lblState.getText().equals("Connected..........")) {
3         String m = "$Restart";
4         out.println(m);
5         out.println(m);
      }
  }
```

داخل زر **Restart** نكتب الكود التالي:

السطر (2): إذا كان الكائن **lblState** مساوي لـــ **"Connected"** أي أنه تم الاتصال عبر الشبكة فنفذ التالي.

السطر (3): عرفنا متغير من نصي **M** والمتغير يقوم بإرسال بيانات للقيام بعمل إعادة تشغيل لجهاز الضحية . وكلمة **$Restart** نكتب داخلها المسار ولكن لن نتطرق لهذا الكود كامل وأنما نأخذ معلومات بيسطة عن كيفية عمل ملفات التجسس.

كود Turn off

```
1 private void btnTurnOffMouseClicked(java.awt.event.MouseEvent evt) {
2     if(lblState.getText().equals("Connected..........")) {
3         String m = "$TurnOff";
4         out.println(m);
5         out.println(m);
      }
  }
```

داخل زر **Turn Off** نكتب الكود التالي:

السطر (2): إذا كان الكائن **lblState** مساوي لـــ **"Connected"** أي أنه تم الاتصال عبر الشبكة فنفذ التالي.

السطر (3): عرفنا متغير من نصي **M** والمتغير يقوم بإرسال بيانات للقيام بعمل إيقاف تشغيل جهاز الضحية.

كود Delete

```
1  private void btnDeleteMouseClicked(java.awt.event.MouseEvent evt) {
2      if(lblState.getText().equals("Connected..........")) {
3          if (jTextField2.getText().equals("")) {
4              JOptionPane.showMessageDialog(null,"Enter File Name");
5          }else{
6              String m = "Delete - " + jTextField2.getText();
7              out.println(m);
8              out.println(m);
9              jTextField2.setText("");
10             jTextField2.requestFocus();
           }
       }
   }
```

داخل زر **Delete** نكتب الكود التالي:

السطر (2): إذا كان الكائن **lblState** مساوي لـــ **"Connected"** أي أنه تم الاتصال عبر الشبكة فنفذ التالي.

السطر (3): إذا كان الكائن **JTextField2** والذي وظيفته إرسال مسار الملف المراد حذفه من جهاز الضحية. فإذا كان المسار في قرص السي لجهاز الضحية فأننا نقوم بكتابة مسار الملف الموجود في قرص السي، كما يلي : **"C:\WINDOWS\Media"**. فأنه سوف يتم حذف الملفات الخاصة بنظام التشغيل ويندوز لجهاز الضحية.

فإذا كان لا يوجد مسار فنقوم بإظهار رسالة تشير إلى أنه يجب كتابة اسم ملف لحذفه، وإذا لم يكن هنالك فراغ أي أنه تم كتابة مسار ملف فنفذ التالي.

السطر (6): تعريف متغير نصي **M** ليأخذ المسار من الكائن **JTextField2** وإرفاقه مع كلمة **" - Delete"** والتي تدل على أن النص الذي بعد هذه الكلمة عبارة عن مسار لملف يجب حذفه.

السطور (7و8): لإخراج المتغير **M** عبر الـــ **Out** والتي وظيفتها كما شرحنا سابقاً بأنها تقوم بعملية الإخراج عبر الشبكة.

السطر (9): جعل الكائن **JTextFiled2** مساوي لفراغ.

السطر (10): إعادة تركيز الماوس على الكائن **JTextFiled2**.

كود Send

```
1  private void btnSendMouseClicked(java.awt.event.MouseEvent evt) {
2      if(lblState.getText().equals("Connected.........")) {
3          if (jTextField1.getText().equals("")) {
4              JOptionPane.showMessageDialog(null,"Enter Message");
5          }else{
6              String m = "Message - " + jTextField1.getText();
7              out.println(m);
8              out.println(m);
9              jTextField1.setText("");
10             jTextField1.requestFocus();
           }
       }
   }
```

داخل زر **Send** نكتب الكود التالي:

السطر (2): إذا كان الكائن **lblState** مساوي لــ **"Connected"** أي أنه تم الاتصال عبر الشبكة فنفذ التالي.

السطر (3): إذا كان الكائن **JTextField1** والذي وظيفته إرسال رسالة إلى جهاز الضحية، فإذا كان فراغ فأنه يجب كتابة رسالة، وإذا كان يوجد نص أي رسالة فنفذ التالي.

السطر (6): تعريف متغير نصي **M** ليأخذ المسار من الكائن **JTextField2** وإرفاقه مع كلمة **" - Message"** والتي تدل على أن النص الذي بعد هذه الكلمة عبارة عن رسالة سيتم إرسالها عبر الشبكة.

السطور (7و8): لإخراج المتغير **M** عبر الــ **Out** والتي وظيفتها كما شرحنا سابقاً بأنها تقوم بعملية الإخراج عبر الشبكة.

السطر (9): جعل الكائن **JTextFiled2** مساوي لفراغ.

السطر (10): إعادة تركيز الماوس على الكائن **JTextFiled2**.

```
private void formWindowOpened(java.awt.event.WindowEvent evt) {
    ConnectTh cth = new ConnectTh();
    cth.start();
}
```

تعريف متغير **cth** من النوع **Connectth** وبداية عمل المتغير داخل الحدث **formWindowOpened** حتى يعمل الاتصال عند فتح الفورم.

أولاً : شاشة FrmServer



<u>سنقوم الآن بشرح الأكواد الخاصة بالمشروع **FrmClient**:</u>

```
1  import java.net.*;
2  import java.io.*;
3  import javax.crypto.NullCipher;
4  import javax.swing.JOptionPane;
5
6  public class frmClient extends javax.swing.JFrame {
7      public frmClient() {
8          initComponents();
9      }
10
11     Socket soc;
12     DataInputStream in;
13     PrintStream out;
```

السطور (1 و 2):
في قسم التصريحات نقوم باستدعاء المكاتب الخاصة بعملية القراءة والكتابة عبر الشبكة.

ثم في قسم التصريحات نعرف كذلك الآتي:
السطر (11): قمنا بتعريف متغير **soc** لنحدد عن طريقه **IP Adress** الخاص بجهاز الضحية وكذلك نحدد البورت الخاص بجهاز السيرفر ليتم التحكم بجهاز الضحية عبر منفذ البورت.
السطر (12و13): تعريف متغيرين الأول **in** لعملية القراءة عبر الشبكة أي الإدخال، والثاني **out** لعملية الكتابة عبر الشبكة أي الإخراج.

ثم في قسم التصريحات نعرف كذلك الآتي:

السطر (1): أنشئنا كلاس من نوع **Thread** وقمنا بتسمية **Connectth** ليقوم بعملية الاتصال عبر الشبكة لجهاز الضحية.

السطر (2): دالة داخلة الكلاس من نوع عام واسمها **run**.

السطر (4): اسندنا قيمة للمتغير **soc** وجعلنا البورت الخاص به يساوي **9000** حيث هذا البورت المنفذ ليتم الدخول عبره من الشبكة، وكذلك نحدد الــ **IP Adress** الخاص بجهاز الضحية ونضع له البورت الخاص بجهاز السرفر (**9000**). وحيث أن بقية الأجهزة المرتبطة بالسرفر يجب أن يكون لها **IP** خاصة لكل جهاز للتعرف عليها ضمن الشبكة ، ما عدا جهاز السرفر فليس له **IP** للتعرف عليه.

السطر (5): نسند قيمة للمتغير **in** ليقوم بإحضار البيانات عبر الشبكة.

السطر (6): نسند قيمة للمتغير **out** ليقوم بإخراج البيانات عبر الشبكة.

ملاحظة مهمة في السطر (7):

لا نقوم بكتابة أي استثناء **Exception** وذلك لأننا لا نريد أن تظهر رسالة خطاء في حالة حدوثه ويعلم بها الضحية، لذلك لا نقوم بكتابة الاستثناء حتى لا تظهر رسائل الخطاء عند حدوث خطاء في الاتصال أو غيرها من الأخطاء.

```
1 public class ConnectTh extends Thread {
2     public void run(){
3         try {
4             soc = new Socket("127.0.0.1",9000);
5             in = new DataInputStream(soc.getInputStream());
6             out = new PrintStream(soc.getOutputStream());
7         }catch(Exception e){}
8     }
9 }
```

```
1 public class ListenTh extends Thread {
2     public void run() {
3         while(true) {
4             try {
5                 if(in.readLine().length() > 1)
6                     NewJob(in.readLine());
7             }catch(Exception e){}
8         }
9     }
10 }
```

ثم في قسم التصريحات نعرف كذلك الآتي:

السطر (1): أنشئنا كلاس من نوع **Thread** وقمنا بتسمية **ListenTh** ليقوم بعملية استقبال الأوامر عبر الشبكة لجهاز الضحية.

السطر (2): دالة داخلة الكلاس من نوع عام واسمها **run**.

السطر (3): جملة **While** حيث أنه طالما أن الــ **Thread** مساوي لــ **True** أي أنه يعمل بطريقة صحيحة فقم بتنفيذ التالي.

السطر (5): إذا كانت الخاصية **readLine** وهي للقراءة عبر الشبكة للمتغير **in** طولها أكبر من واحد أي هنالك نصوص مرسلة عبر الشبكة فقم بتنفيذ الآتي.

السطر (6)نقوم باستدعاء الدالة **NewJob** والتي سنقوم بإنشائها ووظيفتها قراءة النصوص المرسلة للمتغير **in**.

ملاحظة مهمة في السطر (7):

لا نقوم بكتابة أي استثناء **Exception** وذلك لأننا لا نريد أن تظهر رسالة خطاء في حالة حدوثه ويعلم بها الضحية، لذلك لا نقوم بكتابة الاستثناء حتى لا تظهر رسائل الخطاء عند حدوث خطاء في الاتصال أو غيرها من الأخطاء.

```java
1  public void NewJob(String m) {
2      if(m.startsWith("Message - ")) {
3          String x = m.substring(10);
4          JOptionPane.showMessageDialog(null,x);
5      }else if(m.startsWith("Delete - ")){
6          String p= m.substring(9);
7          //Delete code
8      }else if(m.equals("$TurnOff")){
9          //Turn off code
10     }else if(m.equals("$Restart")){
11         //Restart code
12     }
13 }
```

نقوم بإنشاء دالة تحت اسم **NewJob** والتي تقوم باستقبال متغير نصي والذي يحتوي على الأوامر المرسلة عبر الشبكة بشكل نصوص ونجري على التغير النصي العمليات التالية :

السطر (2): إذا كان المتغير **m** بداء بكلمة **Message -** والتي كتبناها في جهاز السيرفر ووظيفتها إظهار رسالة للضحية نقوم بكتابتها نحن في جهاز السيرفر ونرسلها عبر الشبكة بواسطة **Out** إلى الضحية. وحيث أننا نريد فقط نص الرسالة ولا نريد أن تظهر كلمة **Message –** للضحية.

مثال: لو قمنا بإرسال النص التالي عبر الشبكة من جهاز السيرفر :

"Message – Your Computer Will Be Shut Down"

فأننا في السطر (3) :نقتص الكلمة **Message –** كاملة ثم نظهر بقية المتغير **m** والذي يحمل بقية نص الرسالة لكن بعد أن نقتص من البداية وحتى الحرف العاشر والذي ينتهي بعد إشارة (–). أي نقتص 10 حروف وتظهر الرسالة بعد كلمة **Your**.

السطر (4): نقوم بإظهار نص الرسالة المراد إظهار للضحية بعد عملية الاقتصاص لتظهر الرسالة التالية فقط :

(**Your Computer Will Be Shut Down**).

السطر (5): إذا كان المتغير **m** بداء بكلمة **Delete** والتي كتبناها في جهاز السيرفر والتي نكتب بعدها مسار الملف المراد حذفه، فأننا نقوم في السطر (3): بتعريف متغير نصي **p** والذي يقوم باقتصاص 9 حروف من المتغير **m** والذي بدء بكلمة **Delete -** ويأتي بعده كما شرحنا مسار الملف.

وحيث أننا نريد فقط مسار الملف فلذلك قمنا باقتصاص 9 حروف من المتغير **M** فلو فرضنا أن المسار المراد حذفه هو التالي : (**"C:\WINDOWS\Media"**) فأن الرسالة المرسلة عبر الشبكة والتي يقوم بها المتغير **OUT** والذي يحمل لنا كما قلنا سابقاً ما يلي : **"Delete - C:\WINDOWS\Media"**

وبعملية الاقتصاص يتم اقتصاص الكلمة **Delete -** يبقى لنا مسار الملف المراد حذفه فقط وهو التالي:

(**C:\WINDOWS\Media**).

السطر (8): إذا بدء المتغير **m** بكلمة **$Turnoff** ففي السطر (9): المفروض أننا قمنا بكتابة أمر إيقاف تشغيل جهاز الضحية عبر الكود **//Turn Off Code**. ولكن الكود غير موجود وإنما هنا فقط شرحنا برنامج التجسس لندل الطالب على كيفية عمل ملفات التجسس وحتى يعلم كيفية التعالم معها ويعلم بوجودها.

السطر (10): إذا بدء المتغير **m** بكلمة **$Restart** ففي السطر (11): المفروض أننا قمنا بكتابة أمر إعادة تشغيل جهاز الضحية عبر الكود **//Restart Code**. ولكن الكود غير موجود كذلك وإنما شرحنا طريقة عمل ملفات التجسس.

وبعد أن شرحنا الأكراد السابقة بقي علينا القيام بتشغيل الكلاسات الخاصة بالــ **Thread** حتى يعمل الكود في جهاز الضحية.

```
1 private void formWindowOpened(java.awt.event.WindowEvent evt) {
2     //this.hide();
3     ConnectTh cTh = new ConnectTh();
4     cTh.start();
5     ListenTh lTh = new ListenTh();
6     lTh.start();
  }
```

داخل الحدث الخاص ببداية عمل أو فتح الفورم نكتب الكود التالي:

السطر (2):

هنا قمنا بإخفاء الفورم حتى لا يعلم الضحية بوجود نافذة مفتوحة في جهازه أو برنامج تجسس في جهازه.

السطر (3):

نعرف متغير من نوع الكلاس الذي قمنا بإنشائه في نافذة المشروع (الفورم) الخاص بالضحية والذي يبحث عن البورت ونكتب فيه **IP Adress** الخاص بالجهاز والذي كتبنا فيه البورت الخاص بجهاز السيرفر، ونجعل المتغير **cTH** مساوي للكلاس **ConnectTh**.

السطر (4):

تشغيل الكلاس عبر المتغير **cTh**.

السطر (5):

نعرف متغير من نوع الكلاس الذي قمنا بإنشائه في نافذة المشروع (الفورم) الخاص بالضحية والذي يقوم بقراءة البيانات المرسلة عبر الشبكة وينفذ عليها الدالة التي قمنا بكتابتها والمسمى **NewJob** لمعرفة الأمر المرسل هل هو رسالة فقط أم مسار ملف مراد حذفه أم أمر إيقاف تشغيل الجهاز أم أمر إعادة تشغيل جهاز الضحية، ونجعل المتغير **lTH** مساوي للكلاس **ListenTh**.

السطر (6):

تشغيل الكلاس عبر المتغير **lTh**.

قواعد البيانات

درسنا سابقاً في الترم الأول أنه لا يمكن للغة الجافا أن ترتبط بقاعدة بيانات مباشرة مع أي تطبيق مكتوب بهذه اللغة، ولكن يجب علينا أولاً أذا اردنا أن نربط لغة الجافا بقاعدة البيانات فأنه يجب أن نربطها بنظام التشغيل وذلك عبر ما يسمى **Driver** والموجود في نظام التشغيل ويقوم بالربط عن طريق **ODBC** والتي تعني **open DB Connectivity** والذي يقوم بفتح إتصال بقاعدة البيانات والتي سنقوم بتحديدها ونربطها بنظام التشغيل.

تذكير لربط برنامج مصمم بلغة الجافا فأننا نتبع الخطوات التالية:

1- استدعاء مكتبة الربط والمشغلات.
   - أ- استدعاء مكتبة الربط الخاصة بالتعامل مع جمل **SQL**.
   - ب- استدعاء المشغل الخاص بالتعامل مع قواعد البيانات من الجافا.

2- ربط قاعدة البيانات مع البرنامج باستخدام الكائنات.
   - أ- إنشاء رابط (**Connection**).
   - ب- إنشاء جمل التحكم (**Statement**).
   - ت- إنشاء مخزن للبيانات (**Result Set**).

ملاحظة: جميع أوامر قواعد البيانات يجب أن تكتب داخل جملة تصيد الأخطاء وحتى وأن كانت صحيحة.

والآن سنقوم بشرح وإنشاء قاعدة بيانات أكسس.





ننشئ حقلين الأول باسم **Col_no** رقم الكلية من نوع بيانات رقم وعلى أن يكون مفتاح رئيسي .
والحقل الثاني باسم **Col_name** اسم الكلية من نوع بيانات نص



ونقوم بإدخال البيانات التالية إلى الجدول بعد أن قمنا بتصميمه.

والآن سنقوم بربط قاعدة البيانات بنظام التشغيل لتعمل في لغة الجافا.

1- نذهب إلى لوحة التحكم الموجودة في قائمة أبدأ.

2- نذهب ونختار الأداء والصيانة.

Performance and Maintenance

2- نذهب ونختار الأدوات الإدارية.

Administrative Tools

Help

earch Folders Folder Sync

Component Services | Computer Management | Data Sources (ODBC) | desktop.ini | Event Viewer | Local Security Policy | Microsoft .NET Framework ... | Performance | Services

4- نذهب ونختار مصادر البيانات (ODBC).

ODBC Data Source Administrator

User DSN | System DSN | File DSN | Drivers | Tracing | Connection Pooling | About

User Data Sources:

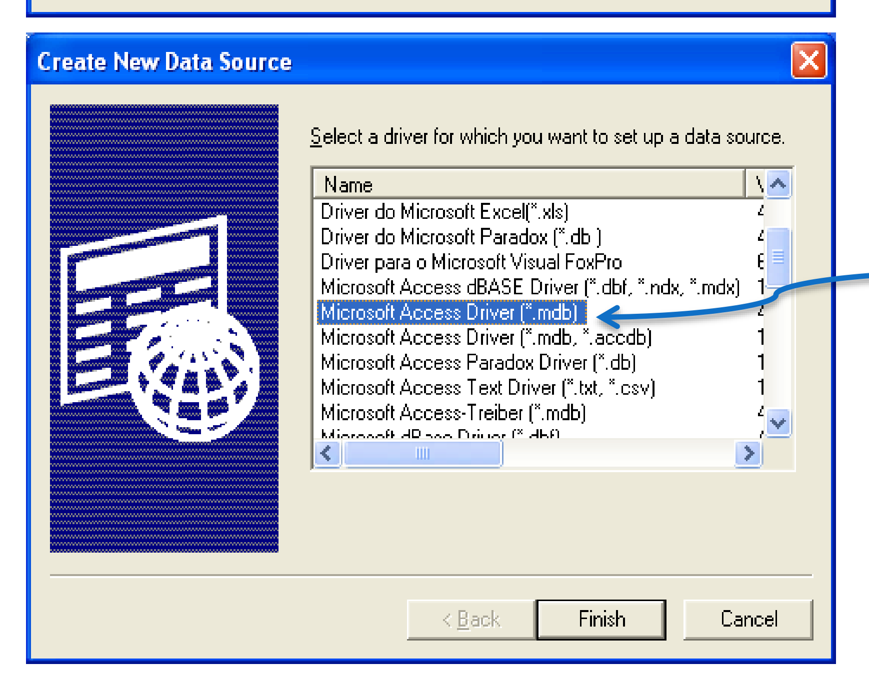
Name	Driver
Col_Link	Microsoft Access Driver (*.mdb)
dBASE Files	Microsoft dBase Driver (*.dbf)
Excel Files	Microsoft Excel Driver (*.xls)
MS Access Database	Microsoft Access Driver (*.mdb)
Visual FoxPro Database	Microsoft Visual FoxPro Driver
Visual FoxPro Tables	Microsoft Visual FoxPro Driver

Add... | Remove | Configure...

An ODBC User data source stores information about how to connect to the indicated data provider. A User data source is only visible to you, and can only be used on the current machine.

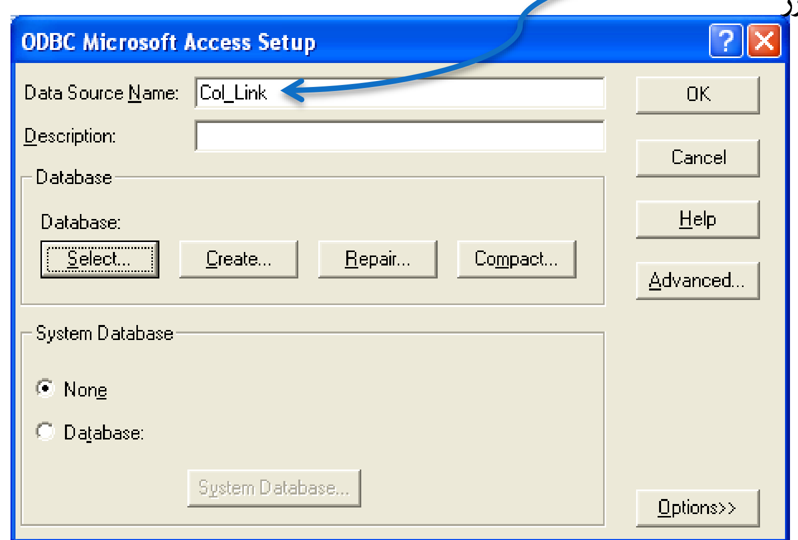
OK | Cancel | Apply | Help

5- تظهر لنا الشاشة التالية ونختار منها إضافة.

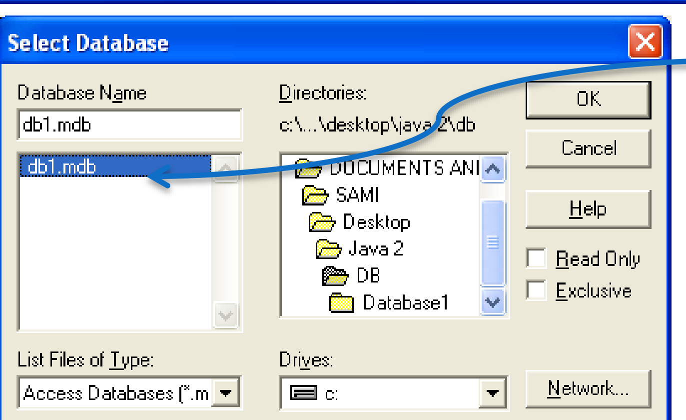


6- تظهر لنا الشاشة التالية ونختار منها نوع قاعدة البيانات (اكسس).

7- تظهر لنا الشاشة التالية ونكتب في المربع اسم الرابط الخاص باسم الكلية ثم نضغط على زر موافق.



7- تظهر لنا الشاشة التالية ونقوم بتحديد مسار قاعدة البيانات في أي مكان داخل جهاز الكمبيوتر ثم نضغط على الزر موافق.



8- ثم سنعود إلى الشاشة التالية وسنرى أن الرابط قد تم انشاءه تحت اسم **Col_Link**.

وهنا قمنا بربط نظام التشغيل بقاعدة البيانات.

ODBC Data Source Administrator
User DSN | System DSN | File DSN | Drivers | Tracing | Connection Pooling | About
User Data Sources:

Name	Driver
Col_Link	Microsoft Access Driver (*.mdb)
dBASE Files	Microsoft dBase Driver (*.dbf)
Excel Files	Microsoft Excel Driver (*.xls)
MS Access Database	Microsoft Access Driver (*.mdb)
Visual FoxPro Database	Microsoft Visual FoxPro Driver
Visual FoxPro Tables	Microsoft Visual FoxPro Driver

Add...
Remove
Configure...
An ODBC User data source stores information about how to connect to the indicated data provider. A User data source is only visible to you, and can only be used on the current machine.
OK
Cancel
Apply
Help

الآن سنقوم بتصميم الواجهة التالية وكتابة الأكراد الخاصة بها لربطها بقاعدة البيانات.



الأكواد :

```
import java.sql.*;
```

في قسم التصريحات :
يجب أن نستدعي المكتبة الخاصة بالتعامل مع جمل الاستعلام **SQL**.

```
1 Connection con;
2 Statement sql;
3 ResultSet rs;
4 String Search ="";
```

في قسم التصريحات :

السطر (1): نعرف متغير **con** ليقوم بعملية الربط والاتصال بقاعدة البيانات التي قمن بإنشاءها.

السطر (2): نعرف متغير **sql** ليقوم بتنفيذ جمل الاستعلام.

السطر (3): نعرف متغير **rs** لخزن نتائج الاستعلام (بيانات) داخل هذا المتغير.

السطر (4): تعريف متغير للبحث من نو ع نص وسنأتي على شرحه لاحقاً.

```java
1  private void ConnDB() {
2      try {
3          Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
4          con = DriverManager.getConnection("jdbc:odbc:Col_link");
5          sql = con.createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,ResultSet.CONCUR_UPDATABLE);
6          rs = sql.executeQuery("Select * from Col_table" + Search);
7
8          DefaultTableModel dtm = new DefaultTableModel();
9          jTable1.setModel(dtm);
10         dtm.addColumn("College no");
11         dtm.addColumn("College name");
12         if(rs.next()){
13             jTextField1.setText(rs.getString("Col_no"));
14             jTextField2.setText(rs.getString("Col_name"));
15             rs.first();
16             do{
17                 Object ary[]= {rs.getInt("Col_no"),rs.getString("Col_name")};
18                 dtm.addRow(ary);
19             }while (rs.next());
20             rs.first();
21         } else {
22             jTextField1.setText("");
23             jTextField2.setText("");
24             jTable1.removeAll();
25         }
26     }catch(Exception e){
27         JOptionPane.showMessageDialog(null,e.getMessage());
28         System.exit(0);
29     }
30 }
```

في قسم التصريحات نقوم بإنشاء الدالة الخاصة بالاتصال بقاعدة البيانات ونسميها **ConnDB** :

السطر (3): **Class** هي مكتبة خاصة بلغة الجافا وكلمة **ForName** هي خاصية تابعة للكلاس ووظيفتها استدعاء **JDBC** لتتم عملية ربط قاعدة البيانات بلغة الجافا.

السطر (4): الاتصال بقاعدة البيانات عن طريق الرابط الذي قمنا بإنشائه والمسمى **Col_Link**.

السطر (5): إنشاء جمل استعلام وتطبيقها داخل المشروع والكلام الذي داخل القوس () هو لكي تتم عملية التنقل بين السجلات ذهاباً وعودة وحتى لا يحصل خطاء أثناء التنقل بين السجلات.

السطر (6): تنفيذ جمل الاستعلام المطلوبة وإسناد نتائجها إلى المتغير **rs**.

السطر (8): نعرف مخزن للجدول باسم **dtm** وحيث أنه لا يتم تعبئة الجدول إلا عن طريق مصفوفة والمصفوفة ستأخذ البيانات من **DB** وتضعها في المتغير **dtm** والذي يظهرها في الجدول.

السطر (9): نجعل الجدول يأخذ بياناته من المتغير **dtm**.

السطر (10و11): نضيف عمودين في المتغير **dtm** والذي سينقل العمودين إلى الجدول ونسمى الأول برقم الكلية والثاني اسم الكلية.

إعداد الطالبان :
سليم شرف ناجي الشميري – هيثم فكري عبد الرقيب العواضي
تحت إشراف الأستاذ:
ماهر عبد الرحمن

السطر (12): بعد جلب البيانات من DB وأسنداها سجل الى المتغير rs فأذا كان هنالك سجل تالي فنفذ التالي.
السطر (13و14): قم بجلب السجل من DB وإظهار رقم الكلية في الكائن JTextField1 واسم الكلية في الكائن JTextField2 .

السطر(15): الذهاب إلى أول سجل لتنفيذ جملة الـــ Do وحيث أنه قمنا بكتابة الـــ Do while بدون أن نكتب rs.first فأنه لن يذهب إلى أول سجل وسيبدأ تنفيذ جملة الـــ Do while من ثاني سجل لأنها تنفذ ثم تفحص الشرط لذلك قمنا نذهب إلى أول سجل لينتقل إلى ثاني سجل ولا يتجاهل أول سجل.
السطر (17): نعرف متغير من نوع عام ليأخذ البيانات الرقمية رقم الكلية والنصية اسم الكلية من DB ويضعها في المتغير ary لكي يقوم المتغير dtm بأخذ البيانات من المصفوفة وإظهارها في الجدول.

السطر (18): نقوم بإضافة سجل للمتغير dtm والذي يأخذ بياناته من المصفوفة والتي تجلب البيانات على سجل سجل من قاعدة البيانات(DB).
السطر (19): سيظل تنفيذ الشرط حتى لا يصبح هنالك سجل تالي عندها يتم إيقاف إسناد قيم للمصفوفة والمتغير dtmوالذي يقوم بملئ الجدول بالبيانات.
السطر (20): العودة إلى أول سجل في قاعدة البيانات.
السطر (21): عند حالة عدم تنفيذ جملة الـــ IF أي أنه لا يوجد سجل تالي اذهب ونفذ التالي.
السطر (22و23): اجعل الكائنين JTextFiled1 ,2 فارغين.
السطر (24): جعل الجدول فارغ من أسماء الأعمدة حيث أنه عند تنزيل الجدول يضع أربعة أعمدة فلذلك نقوم بمسحها ونجعل قيم الجدول من المتغير dtm.
السطر (28و29): إظهار رسالة في حالة وجود خطاء في الاتصال بقاعدة البيانات والخروج من التطبيق.

```
private void formWindowOpened(java.awt.event.WindowEvent evt) {
    ConnDB();
}
```

في الحدث الخاص بفتح أو بداية عمل الفورم نقوم باستدعاء الدالة الخاصة بالاتصال بقاعدة البيانات.

كود New

```
1  private void btnSaveMouseClicked(java.awt.event.MouseEvent evt) {
2      if(btnSave.getText().equals("New")) {
3          jTextField1.setEnabled(true);
4          jTextField2.setEnabled(true);
5          jTextField1.setText("");
6          jTextField2.setText("");
7          btnSave.setText("save");
8      } else if(!(jTextField1.getText().equals("")) && !(jTextField2.getText().equals("")) ) {
9          try {
10             String s = "insert into Col_table values(?,?)";
11             PreparedStatement ps = con.prepareStatement(s);
12             ps.setInt(1,Integer.parseInt(jTextField1.getText()));
13             ps.setString(2,jTextField2.getText());
14             int t = ps.executeUpdate();
15             if (t > 0) {
16                 JOptionPane.showMessageDialog(null,"Saved....");
17                 ConnDB();
18                 jTextField1.setEnabled(false);
19                 jTextField2.setEnabled(false);
20             }
21             btnSave.setText("New");
22         }catch(Exception e){
23             JOptionPane.showMessageDialog(null,e.getMessage());
24         }
25     } else {
26         JOptionPane.showMessageDialog(null,"Enter values");
27     }
28 }
```

السطر (2): إذا كان الزر الحفظ مساوي لجديد **New** فنفذ التالي.
السطر (3و4و5و6): أجعل الكائنين **JTextFiled1,2** مفعلة وقم بجعل الاسم الظاهري لهما مساوي لفراغ.
السطر (7): أجعل الاسم الظاهري للكائن **btnSave** يساوي **Save**
السطر (8): وإذا لم يكن الاسم الظاهري للزر حفظ مساوي لجديد قم بتنفيذ جملة الــ **If** الشرط الأخرى والتي تدل على أنه إذا كان الكائنين **JTextFiled1,2** لا يسوي فراغ فنفذ التالي.
السطر (10): نعرف متغير **s** من نوع نصي ونسند له قيمة والتي هي عبارة عن جملة استعلام نصية وفحواها إدخال بيانات إلى جدول الكلية.
السطر (11): تمرير قيم الاستعلام الموجودة داخل المتغير **s** إلى المتغير **ps**.
السطر (12و13): الارقام **1** و **2** تدل على قيم الموجودة بالتوالي في المتغير **S** ؟ تأخذ رقم واحد و ؟ الثانية تأخذ رقم أثنين. وهذين السطرين تحتوى على جمل خاصة بعملية تمرير القيم واستبدال علامة الاستفهام الموجودة في المتغير **S** بقيمة في جملة الاستعلام.
السطر (14): نعرف متغير **t** من نوع رقم ونسند له قيمة وهي عبارة عن الرقم الممر في حالة تنفيذ الاستعلام حيث إذا نفذ الاستعلام فأنه سيتم تمرير الرقم واحد والذي يدل على تم تنفيذ الاستعلام والا سيتم تمرير الرقم صفر والذي يدل على أن الاستعلام لم ينفذ.
السطر (15): إذا كان المتغير **t** أكبر من الصفر أي تم تنفيذ الاستعلام وتم إضافة بيانات فنفذ التالي.
السطر (16): إظهار رسالة بأنه تم عملية الحفظ.
السطر (17): استدعاء الدالة أو الاجراء الخاص بالاتصال ليتم إضافة البيانات الجديدة داخل المشروع.
السطر (18و19): جعل الخاصية تفعيل الكائنين **JTextFiled1,2** مساوية لــ **False**.
السطر (21): جعل الاسم الظاهري للزر **btnSave** يساوي جديد **New**.

كود Update

```java
1  private void btnUpdateMouseClicked(java.awt.event.MouseEvent evt) {
2      if (btnUpdate.getText().equals("Update")) {
3          jTextField1.setEnabled(true);
4          jTextField2.setEnabled(true);
5          btnUpdate.setText("Save");
6      } else {
7          try {
8              String s= "Update Col_table set Col_name = ? " +
9                      " Where Col_no = ?";
10             PreparedStatement ps = con.prepareStatement(s);
11             ps.setString(1,jTextField2.getText());
12             ps.setInt(2,Integer.parseInt(jTextField1.getText()));
13             int t = ps.executeUpdate();
14             if(t > 0) {
15                 JOptionPane.showMessageDialog(null,"Updated......");
16                 ConnDB();
17             }
18         }catch(Exception e){
19             JOptionPane.showMessageDialog(null,e.getMessage());
20         }
21     }
22 }
```

السطر (2): إذا كان الزر تعديل مساوي لجديد **Update** فنفذ التالي.

السطر (3و4): أجعل الكائنين **JTextFiled1,2** مفعل.

السطر (5): أجعل الاسم الظاهري للكائن **btnUpadte** يساوي **Save**.

السطر (8): نعرف متغير **s** من نوع نصي ونسند له قيمة والتي هي عبارة عن جملة استعلام نصية وفحواها تعديل بيانات جدول الكلية بدلالة المتغير (؟) رقم الكلية.

السطر (10): تمرير قيم الاستعلام الموجودة داخل المتغير **s** إلى المتغير **ps**.

السطر (11و12): الأرقام **1** و **2** تدل على قيم الموجودة بالتوالي في المتغير **S** (؟) تأخذ رقم واحد و (؟) الثانية تأخذ رقم أثنين. وهذين السطرين تحتوى على جمل خاصة بعملية تمرير القيم واستبدال علامة الاستفهام الموجودة في المتغير **S** بقيمة في جملة الاستعلام.

السطر (13): نعرف متغير **t** من نوع رقم ونسند له قيمة وهي عبارة عن الرقم الممر في حالة تنفيذ الاستعلام حيث إذا نفذ الاستعلام فأنه سيتم تمرير الرقم واحد والذي يدل على تم تنفيذ الاستعلام وإلا سيتم تمرير الرقم صفر والذي يدل على أن الاستعلام لم ينفذ.

السطر (14): إذا كان المتغير **t** أكبر من الصفر أي تم تنفيذ الاستعلام وتم إضافة بيانات فنفذ التالي.

السطر (16): إظهار رسالة بأنه تم عملية التعديل.

السطر (17): استدعاء الدالة أو الإجراء الخاص بالاتصال ليتم إضافة البيانات الجديدة داخل المشروع.

السطر (18): إظهار رسالة الخطاء في حالة حدوث خطاء.

كود Delete

```
1  private void jButton2MouseClicked(java.awt.event.MouseEvent evt) {
2      if(jTextField1.getText().equals("")) {
3          JOptionPane.showMessageDialog(null,"Cant delete Filed");
4      }else{
5          try {
6              String s = "Delete from Col_table where Col_no = ?";
7              PreparedStatement ps = con.prepareStatement(s);
8              ps.setInt(1,Integer.parseInt(jTextField1.getText()));
9              int t= ps.executeUpdate();
10             if (t > 0 ) {
11                 JOptionPane.showMessageDialog(null,"Deleted........");
12                 ConnDB();
13             }
14         }catch(Exception e){
15             JOptionPane.showMessageDialog(null,e.getMessage());
16         }
17     }
18 }
```

السطر (2): إذا كان الكائن JTextFiled1 والذي يظهر رقم الكلية مساوي لفراغ.
السطر (3): إظهار رسالة بأنه لا يوجد رقم للكلية للقيام بعملية الحذف.
السطر (6): نعرف متغير s من نوع نصي ونسند له قيمة والتي هي عبارة عن جملة استعلام نصية وفحواها حذف بيانات جدول الكلية بدلالة المتغير (؟) رقم الكلية.
السطر (7): تمرير قيم الاستعلام الموجودة داخل المتغير s إلى المتغير ps .
السطر (8): الرقم 1 يدل على أن المعامل (؟) الموجود في المتغير S يأخذ رقم الكلية. وهذا السطر تحتوى على جملة خاصة بعملية تمرير القيم واستبدال علامة الاستفهام الموجودة في المتغير S بقيمة في جملة الاستعلام وهي قيمة علمية الحذف بدلالة رقم الكلية.
السطر (9): نعرف متغير t من نوع رقم ونسند له قيمة وهي عبارة عن الرقم الممر في حالة تنفيذ الاستعلام حيث إذا نفذ الاستعلام فأنه سيتم تمرير الرقم واحد والذي يدل على تم تنفيذ الاستعلام وإلا سيتم تمرير الرقم صفر والذي يدل على أن الاستعلام لم ينفذ.
السطر (10): إذا كان المتغير t أكبر من الصفر أي تم تنفيذ الاستعلام وتم إضافة بيانات فنفذ التالي.
السطر (11): إظهار رسالة بأنه تم عملية حذف السجل بدلالة رقم الكلية.
السطر (12): استدعاء الدالة أو الإجراء الخاص بالاتصال ليتم إضافة البيانات الجديدة داخل المشروع.
السطر (14): إظهار رسالة الخطاء في حالة حدوث خطاء.

كود Search

```
1 private void btnSearchMouseClicked(java.awt.event.MouseEvent evt) {
2     if (jTextField3.getText().equals("")) {
3         Search="";
4         ConnDB();
5     } else {
6         Search = " Where Col_name like '%" + jTextField3.getText() + "%'";
7         ConnDB();
8     }
9 }
```

السطر (2): إذا كان الكائن JTextFiled3 والذي نكتب فيه أي نص أو رقم نريد البحث عنه داخل سجلات قاعدة البيانات.
السطر (3): إسناد قيمة للمتغير الخاص بعملية البحث ويساوي فراغ.
السطر (4): استدعاء الدالة أو الإجراء الخاص بالاتصال ليتم إضافة البيانات الجديدة داخل المشروع.
السطر (5): إذا كان الكائن JTextFiled3 لا يساوي فراغ أي أن هناك نص تم كتابته أو ربما رقم فسوف يقوم بتنفيذ التالي.
السطر (6): لو لاحظنا أنه إثناء إنشاء جملة الاستعلام في الاجراء ConDB
(rs = sql.executeQuery("Select * from Col_table" + Search);)
وهو ليقوم بإضافة المتغير Search إلى جملة الاستعلام rs كما لاحظنا في الكود السابق والذي قمنا بكتابته عند كتابة نص الاتصال بقاعدة البيانات داخل الاجراء ConDB ولتصبح جملة الاستعلام كالتالي:

Select * from Col_table Where Col_name like '%" + jTextField3.getText() + "%'";

السطر (7): استدعاء الدالة أو الإجراء الخاص بالاتصال ليتم إضافة البيانات الجديدة داخل المشروع وتحديث نافذة المشروع.

كود <<

```
1  private void btnLastMouseClicked(java.awt.event.MouseEvent evt) {
2      try {
3          if(rs.last()) {
4              jTextField1.setText(rs.getString("Col_no"));
5              jTextField2.setText(rs.getString("Col_name"));
6          }
7      }catch(Exception e){
8          JOptionPane.showMessageDialog(null,e.getMessage())
9      }
10 }
```

السطر (3): إذا كان المتغير rs والذي يحتوي على جملة الاستعلام قد وصل الى آخر سجل فقم بتنفيذ التالي.
السطور (4و5): اسند قيمة للكائنين JTextFiled1,2 قيم المتغير rs لأول عمود في قاعدة البيانات وثاني عمود.
السطر (7): تنفيذ جملة الاستثناء في حالة وجود خطاء.

كود <

```
1  private void btnNextMouseClicked(java.awt.event.MouseEvent evt) {
2      try {
3          if(!(rs.isLast()) && rs.next()) {
4              jTextField1.setText(rs.getString("Col_no"));
5              jTextField2.setText(rs.getString("Col_name"));
6          }
7      }catch(Exception e){
8          JOptionPane.showMessageDialog(null,e.getMessage())
9      }
10 }
```

السطر (3): إذا كان المتغير rs والذي يحتوي على جملة الاستعلام لا يساوي آخر سجل وكذلك هنالك سجل تالي فقم بتنفيذ التالي.
السطور (4و5): اسند قيمة للكائنين JTextFiled1,2 قيم المتغير rs لأول عمود في قاعدة البيانات وثاني عمود.
السطر (7): تنفيذ جملة الاستثناء في حالة وجود خطاء.

كود >

```
1  private void btnPreviousMouseClicked(java.awt.event.MouseEvent evt) {
2      try {
3          if(rs.previous()) {
4              jTextField1.setText(rs.getString("Col_no"));
5              jTextField2.setText(rs.getString("Col_name"));
6          }
7      }catch(Exception e){
8          JOptionPane.showMessageDialog(null,e.getMessage());
9      }
10 }
```

السطر (3): إذا كان المتغير rs والذي يحتوي على جملة الاستعلام ذهب إلى سجل السابق وهنالك سجل سابق فقم بتنفيذ التالي.
السطور (4و5): اسند قيمة للكائنين JTextFiled1,2 قيم المتغير rs لأول عمود في قاعدة البيانات وثاني عمود.
السطر (7): تنفيذ جملة الاستثناء في حالة وجود خطاء.

كود >>

```
1  private void BtnFirstMouseClicked(java.awt.event.MouseEvent evt) {
2      try{
3          if(rs.first()) {
4              jTextField1.setText(rs.getString("Col_no"));
5              jTextField2.setText(rs.getString("Col_name"));
6          }
7      }catch(Exception e){
8          JOptionPane.showMessageDialog(null,e.getMessage());
9      }
10 }
```

السطر (3): إذا كان المتغير rs والذي يحتوي على جملة الاستعلام هو أول سجل فقم بتنفيذ التالي.
السطور (4و5): اسند قيمة للكائنين JTextFiled1,2 قيم المتغير rs لأول عمود في قاعدة البيانات وثاني عمود.
السطر (7): تنفيذ جملة الاستثناء في حالة وجود خطاء.

